Brno University of Technology at TRECVid 2008

Petr Chmelař, Vítězslav Beran, Adam Herout, Michal Hradiš, Roman Juránek,
Aleš Láník, Jozef Mlích, Jan Navrátil, Ivo Řezníček, Pavel Žák, Pavel Zemčík

Brno University of Technology

Faculty of Information Technology
Božetěchova 2
Brno, 612 66
Czech Republic

Abstract

In this paper we describe our experiments in all task of TRECVid 2008. This year, we have concentrated mainly on the local (affine covariant) image features and its transformation into a search-able form for the Content-based copy detection pilot together with the indexing and search techniques for the Search task and a practical test of the background subtraction and trajectory generation algorithms for the Surveillance pilot.

In brief, we have submitted the following tasks:

1. Surveillance event detection pilot. We have participated in the detection of the following events – PersonRuns, ObjectPut, ElevatorNoEntry and OpposingFlow. It has been based mainly on advanced masking and background subtractions and extracted trajectories.
2. Content-based copy detection pilot. We have submitted one run based on search of the joint image features - global (color, texture) and local features (SIFT).
3. High-level feature extraction. We have used two training methods based on SVM using color, texture and face image features. First only selected subset of the training data, second all the annotated data were used for the training.
4. Search. We have performed two fully automatic IR experiments based on the text of the queries and ASR/MT provided by NIST and the data consumed by the High-level feature extraction task.
5. Rushes summarization, to which is dedicated a separate paper [2].

The paper is organized as follows. In section 1, the overview of the Brno University of Technology, Faculty of information Technology and participant groups are described. The surveillance event detection task is significantly different to the others so we dedicate the chapter 2 to the task. The details of other tasks submitted are given in section 3. Chapter 4 discusses the achieved results and concludes the paper.

1. Introduction

The Brno University of Technology, Faculty of Information Technology has taken apart at TRECVid second time this year [1]. The mayor of the work belongs to the Graph@FIT group.

The university was established as a german-czech technical school in 1949, which turned into the Czech Technical University, founded in 1899. At present, the Brno University of Technology with more than 22,000 students covers the whole spectrum of technical disciplines, eg. in all kinds of engineering as aerospace industry, (bio)computing, architecture and design. One of the most important aims is to accumulate knowledge and apply it for practical purposes.

The Faculty of Information Technology (FIT) has been established in 2002 when the Faculty of Electrical Engineering and Computer Science has split into FIT and the Faculty of Electrical Engineering and Communication. The faculty offers bachelor, master and doctoral study programs in the former Cartesian monastery, founded in 1375 and new buildings in the vicinity. The faculty consists of four departments. Two of them has taken apart in the TRECVid tasks. It is the Graph group – part of Department of Computer Graphics and Multimedia (DCGM) and the Data Mining group – part of Department of Information Systems (DIFS).

Figure 1. The faculty at night.

The Graph@FIT focuses its research on development and application of algorithms for image and video processing and computer vision. The group is responsible for teaching of Image processing and Computer vision courses in masters program at FIT and participates in the research of several projects, such as EU IST CareTaker or AMIDA projects and many national projects. The research areas of Graph@FIT include automatic video editing and summarization, algorithms for detection and tracking of objects and human body parts, acceleration of image processing and computer vision in hardware, detection and positioning of human face and other objects, industrial and traffic applications of image processing, evaluation of trajectories and complex video events and actions.

Further information on the groups, publications, research projects and contacts can be found at http://www.fit.vutbr.cz.

2. Surveillance event detection pilot

Basically, we have used two approaches, first a background detection and subtraction, as described in the [9] and the OpenCV [19] blobtrack. There were many burdens using these tools and techniques.

2.1. Video processing and background subtraction

The principle of the background detection is that the background is modeled as an average value of color in each pixel of video and the foreground is a value different to the background. We have been inspired by the approach described in [9] is based on segmentation of the color space in RGB color space into background, foreground and noise (reflection, shadow, ghost and fluctuation) using a cone with the in the beginning of the RGB coordinate system. In this way the illumination can be separated from the color more easy. However the selection of appropriate parameters is a burden task.

ObjectPut

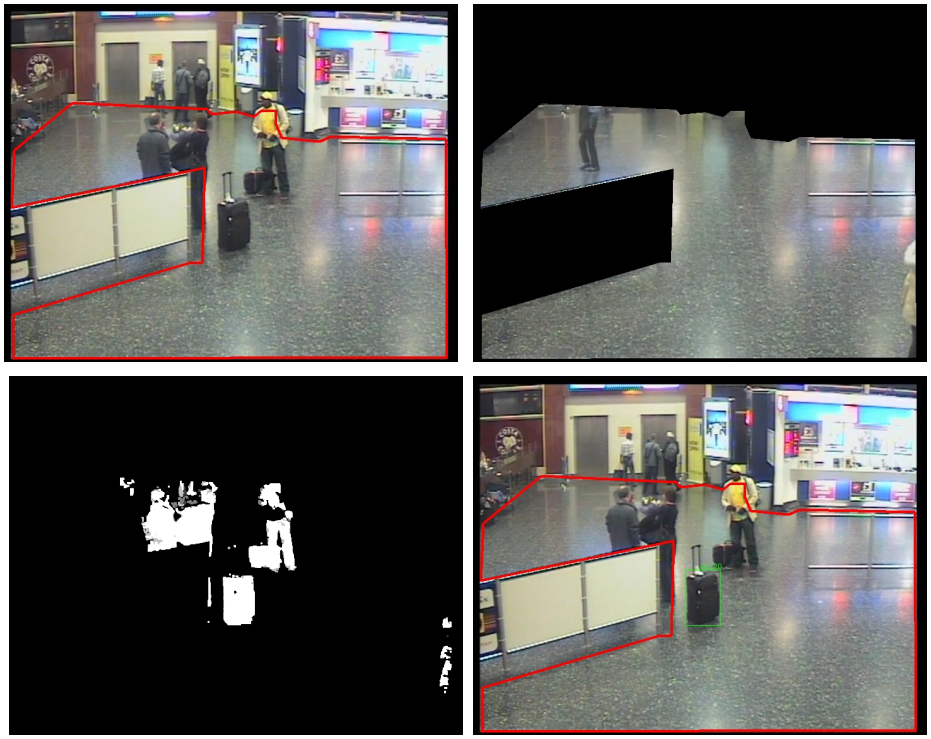


Figure 2. The illustration of the ObjectPut event detected.

The detector has been based on a simple idea, that the left object is supposed not being moving. Based on this presumption, we got a mask of moving objects and used an AND operation of a sequence of several foreground masks. This masks we have segmented in particular regions and those were stored in an active object list. The list was used to produce the output.

2.5. Trajectory-based events

Object tracking and trajectory classification is a very complex problem. Discussed approach is based on well known methods of object tracking [19] and investigates potential of a few methods of trajectory analysis. The tracking was performed on data obtained from static camera, therefore was used method based on Background subtraction. The accuracy of tracking in case of object occlusion was improved by

Kalman filtering. The tracking process is illustrated by Figure 3.

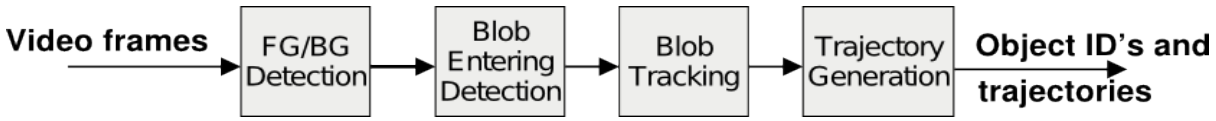


Figure 3. The scheme of the blob tracking module.

Because tracking is a difficult task and the output of a simple tracker even using Kalman filter is usually very noisy, it is good to make some preprocessing to improve the accuracy of the input trajectories. In the following text there are distinguished events determined according to single trajectory and events detected by trajectory pairs and by more than two trajectories.

Elevator no Entry

Door state detection was a part of Elevator No Entry event detection. It is first processing part, it depends on a well selected regions in the top part of elevator's door (always visible). This approach improves robustness of door-stage detector. There are 2 elevators, regions have been prepared for each one. The detector can do two things, first is detection of fully opened door, which uses three independent regions. For each region there is determined an interval of average value of the whole region. All the three regions average values must satisfy defined color interval. Second is a detection of lift door stage, which uses position of the moving region. In the second case, set of regions average values must be computed and compared to the defined interval. Position of the door is defined by an index of computed region which satisfy interval condition and all of this could be converted into percentile.

The door detector marks interesting areas in video, where Elevator no Entry event may happen. These pieces of video serves as input to trajectory-based decision algorithm. Second input to this algorithm is set of moving object's trajectories, which are processed only on selected pieces of video, then reduction of computational time is eminent. Tracking algorithm produces set of trajectories for every piece of video, these trajectories have to be filtered and if Elevator no Entry event occur, corresponding trajectory must be found in this set of trajectories.

Trajectory processing is done in several stages:

1. Non-relevant ugly trajectories, which describe no object, have to be removed from set of trajectories, these trajectories are detected by it's moving area which is tresholded.
2. Second filtering idea is based on fact, that if Elevator No Entry event ocurrs, some person must be in front of elevator before elevator starts opening door, so trajectory, which started after this is non-relevant and could be removed from the set. This approach removes cases, where persons only steps out the elevator.
3. Elevator No Entry event occurs most frequently, that person fetch the elevator, think that it takes long time and all-time is visible in front of elevator. So if trajectory's end occur before closing time of elevator, is non-relevant too and should be removed from the set.
4. If at least one trajectory remains in the set Elevator no Entry event occurs.

Person Runs and Opposing Flow

Detection of the PersonRuns and OpposingFlow events is based only at further processing of the extracted trajectories. Each trajectory is given as a set of the blob size and position in several adjacent time steps. For both events the common scheme of processing is done in the following steps (similarly):

1. Wrong trajectory removal.
2. Trajectory smoothing and feature computing.
3. Candidate trajectories retrieval.
4. Event decision.

First step is simple preprocessing, where the area of the trajectory and maximum and minimum size of blobs is computed. If these information are outside predefined thresholds, the trajectory is considered as unwanted and is removed from any other processing.

Second step divides the trajectory to greater time intervals and additional trajectory behaviour is examined. For each interval the mean position and blob size is computed together with trajectory curvature, average blob speed and main direction of movement withing the given time interval. Based on this information the final event decision could be made. But to reduce the fail positive rate, yet another processing step is needed. Only "well-behaved" parts of trajectories are taken into following decision, that means that only those time intervals whose curvature is low and blob size does not change rapidly are interesting for event detection.

Final step is simple checking if the main event conditions are fulfilled. For PersonRuns the speed above some threshold is checked (where the threshold depends on the camera view and blob position within the frame), for OpposingFlow the direction of movement in specified area of view is observed. All used thresholds are implementation dependent and have been set experimentally.

The event could be described as a pattern which is detected by Gaussian Mixture Model or more general by Hidden Markov Model in part of trajectory. The training phase is done once for each camera. First, we define all trajectory classes corresponding to events in the scene. Secondly, initial models are defined for each trajectory class. Next, the trajectory classes in video sequence are annotated and models are adapted accordingly (training process). For better model estimation in the training phase it is more suitable to use the most representative trajectories. The classification step produces evaluation of incoming trajectories. Degree of correspondence of the trajectory with model is expressed by likelihood. According to the this, the event with the best result is selected.

The approach described in [6] could detect also events with small training sets by determining and describing of all usual behaviors in scene and assumption, that the trajectory of wanted event is significantly different then trajectories of normal behavior. For a successful classification of trajectories and for search of abnormal trajectories, it is important to have a well defined scene with well defined scenarios, because anomalous trajectories correspond to scenarios that are not defined.

The events which could be detected only according to two or more trajectories could be analyzed in the same way as events mentioned above. However, the computational cost is higher. Examples of such events are (embrace, people met, people splits). Several issues can arise when analyzing trajectory pairs. In the case the analysis involves more than two trajectories, the issues mentioned before become even more problematic and we haven't submitted such runs, however we have been working on it.

YellowVest

Trajectory processing for YellowVest event detection does not need to be much sophisticated, because the trajectory extraction subsystems performs well for yellow vest tracking. So only color check is done several times during relevant time interval and if the area of detected blobs is mostly yellow colored in all checks, the YellowVest event is marked as the trajectory duration. The color check is done in RGB space where for each camera the average color is trained and color the comparison is based on the same model as in background subtraction module. We haven't submitted any results for this event.

3. Sound and Vision dataset based tasks

The Content-based copy detection pilot, High-level feature extraction and the Search tasks have many in common, thus they are presented altogether. Before the particular tasks are described, the low-level feature extraction, face detection, clustering and indexing, search techniques used are described.

3.1. Feature extraction

The low-level features used for frame description utilized in the system are color histogram based on HSV color model and multi-scale gradient distribution of a frame intensity.

Color histogram Based on HSV color model

The color histogram contains statistical information about color distribution in terms of frequency of hues and saturations in the frame (using HSV color model). The better spatial description is achieved by dividing the frame into several patches. The frame division is not adaptive, so the patches have a similar size. Each patch is processed separately; the histogram is computed and normalized.

Multi-scale gradient distribution

The histogram of gradient orientations serves as other part of the feature vector. First, the frame gradients are computed. Then each gradient contributes to the histogram bin according to its orientation. The contributions are weighted by the gradient magnitude. The gradients are computed on different frame resolutions so also lower frequency structures contribute to final feature vector. The resolution of both the color histogram and gradient histogram, the resolution of the frame grid and amount of frame scale levels are all the descriptor parameters.

Color layout

The color layout computation is based on the JPEG compression technique. First, the image is resampled into 8x8 pixels in Y'CbCr color model. Then, the discrete cosine transform (DCT) is applied on each channel. The descriptor coefficients are then extracted zig-zag [12], as illustrated in figure 4. We use 20 (Y) + 15 (Cb and Cr) coeffs, thus the feature vector has 50 coefficients.

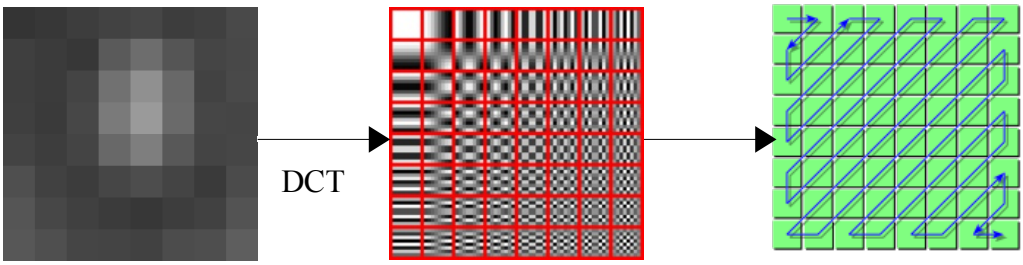
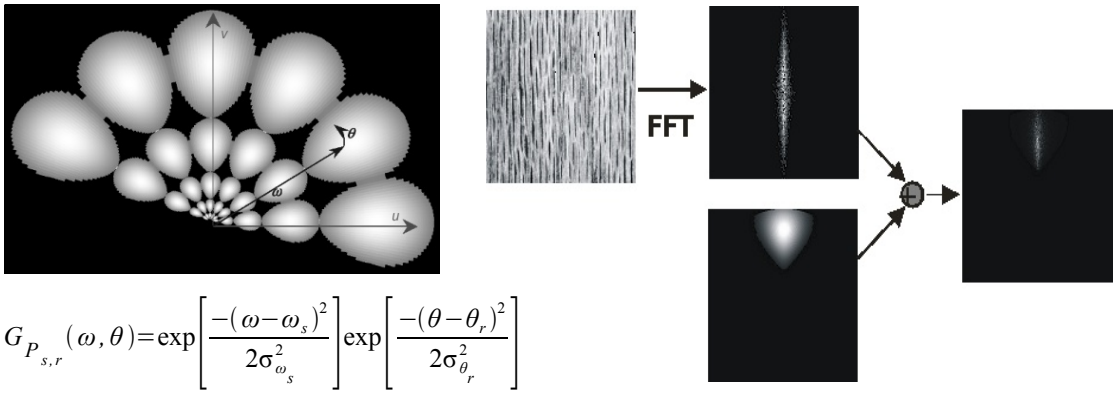


Figure 4. Illustration of the color layout description process.

Gabor texture

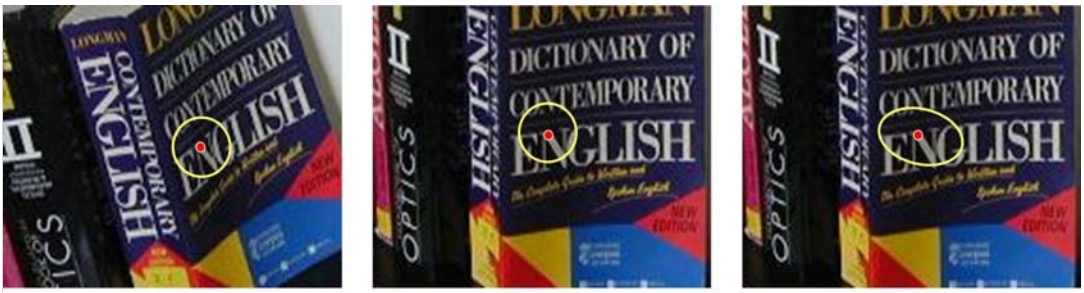
Using a bank of Gabor filters [13] in the frequency domain, we can divide the the space, created eg. using Fourier transform, into bands, as illustrated in figure 5. We use the first moments of energy in the filtered 30 sub-bands (G_P) – 6 angular (*30°*) and 5 radial (in octaves) for construction of the descriptor.



$$G_{P_{s,r}}(\omega,\theta)=\exp\left[\frac{-(\omega-\omega_s)^2}{2\sigma_{\omega_s}^2}\right]\exp\left[\frac{-(\theta-\theta_r)^2}{2\sigma_{\theta_r}^2}\right]$$

Figure 5. Illustration of the Gabor texture description process.

Local features



(a) (b) (c)

Figure 6. Class of transformations needed to cope with viewpoint changes. (a) First viewpoint; (b,c) secondviewpoint. Fixed size circular. Patches (a,b) clearly do not suffice to deal with general viewpoint changes.What is needed is an anisotropic rescaling, i.e., anaffnity (c) [14].

(a) (b) (c) (d) (e)

Figure 7. Affine covariant regions offer a solution to viewpoint and illumination changes. First row: one viewpoint; second row: other viewpoint. (a) Original images, (b) detected affine covariant regions, (c) close-up of the detected regions. (d) Geometric normalization to circles. The regions are the same up to rotation. (e) Photometric and geometric normalization. The slight residual difference in rotation is due to an estimation error [14].

Maximally Stable Extremal Regions

Or MSER in short, we have used for finding of connected components of an appropriately thresholded image (experimentally) to be maximally stable. Extremal means that all the pixels inside the region have intensity lower (darker) or higher than pixels on the edge of the region [14].

Scale Invariant Feature Transform

Or SIFT in short [14] we use for description of regions found by MSER. It captures an information (about an elliptic neighborhood) of the point of interest (center of the region) using a histogram of locally oriented gradients and stores it as a vector of size 128 (8 orientations, each in 4x4 locations).

Speeded Up Robust Features

Or SURF in short is used for the detection of interesting points based on determination of determinant of Hessian matrix (second partial derivations) of an integral image. Description is based on Haar transformation, similarly to the SIFT descriptor..

Face detection

Knowledge about presence of people in a video is a valuable source of information for both Search and High-level feature extraction tasks. Many possible ways of detecting people in image exist, one of which is the face detection. The face detection task has been well studied and many methods for reliable real-time detection exist. The most successful are appearance-based methods which use some variation of a cascade of simple boosted classifiers to scan the images. This approach was originally proposed by Viola and Jones in 2001 [20].

We have used a frontal face detector to extract four low-level features from the video frames. One of the features was the total number of faces present in a video frame. The other three features were the numbers of small, medium and large faces in the frame. The rationale behind this choice was that the number and sizes of faces are more informative then their positions and that the number of detections is too small to make this information even sparser by considering the position.

Total four classifiers with random initial choice of training samples were created by the WaldBoost algorithm [FACE2]. During detection, the responses of the individual classifiers were integrated by weighted voting to get the final detections. Instead of the traditional Haar-like features, the Local Rank Patterns (LRP) [4] were used as features by the classifiers. The LRP features were chosen because they show better performance on the frontal face detection task [4] then the Haar-like features especially in combination with the slower and more precise classifiers which were used for this particular task. The classifiers were created using our experimental framework for research on detection classifiers [5].

3.2. Indexing and search techniques

For the content-based copy detection and search tasks, before all, we have used the PostgreSQL database system. There we store all extracted features, video and shot metadata, annotations, ASR and MT data.

For the fixed-length (low-level) features, we have employed standard Eukeidean distance ($p=2$ in) $d(p_1, p_2) = \left(\sum_{i=1}^{n} \left(p_1[i] - p_2[i] \right)^p \right)^{\frac{1}{p}}$. For the variable-length ones the cosine distance $r(d_q, d_d) = \frac{d_q \cdot d_d}{|d_q||d_d|}$. In the cosine distance the (words) weighting is performed using TF-IDF: *tf-idf(w) = tf(w)idf(w)*, where $tf(w) = \frac{|d(w)|}{|d|}, \quad idf(w) = \log\left(\frac{|D|}{|D(w)|} \right)$. We also use the Generalized Inverted (document) Index (GIN, [16]) to speed up the queries.

However, these techniques were unable to find anything in case of the local features in serious time, thus we had to employ some reduction of the search space, similarly to the [18]. However, in that approach the dictionary construction (clustering) is even much more severe, because the search can be parallelized.

Thus we have used another one described next.

Voronoi tessellation based clustering of the local features

Although, there are many clustering techniques, it is not possible to use them for all purposes. The initiative problem was to create as many clusters as possible (eg. thousands) for the local image features description in huge amount of video for Content-based copy detection and Search tasks. We had 25 mil. of MSER/SIFT [14] feature vectors (32 GB) and 38 milion SURF [15] descriptors (41 GB) of local invariant features, as described in section 3.1. These large dimensional vectors cover the space almost continuously and commonly used clustering methods are unable to create enough classes or to finish in serious time. For that purpose we have used (also modified) versions of KMeans and DBSCAN clustering methods [8]. However in the related literature, there is described the problem to have solution only for approximately 1GB of the data, when thousands of classes needed.

Therefore, we have invented a new method based on Voronoi tessellation [7] that needs no more than two passes through the data. The approach is based on discovery of clusters in higher density locations. Because of large dataset, it is possible to create higher amount of candidate clusters and select appropriate number of classes (large but not huge) and the rest data assign to these classes. The method has been implemented as a set of SQL functions and queries, the pseudo code follows:

```
Algoritm 1: Candidate classes discovery.
for each (SELECT f.id, f.features FROM lf_table as f) { // rand
     SELECT c.id, distance(f.features, features) AS dist
        FROM lf_clusters AS c
        ORDER BY dist LIMIT 1;
    if (dist > treshold) // log2(|f|^avg(stdev(f[i])))
        INSERT INTO lf_clusters VALUES (f.id=next(), features);
    UPDATE lf_table SET cluster=c.id WHERE id=f.id;
} // if(SELECT COUNT(*) FROM lf_clusters > 10*classes) break; -- optimization

Algoritm 2: Two variants of class selection (third is using a clustering method :)
a) DELETE FROM lf_clusters WHERE id IN (
       SELECT cluster, count(id) AS cnt FROM lf_table
       GROUP BY cluster HAVING cnt < min_objects )
b) DELETE FROM lf_clusters WHERE id NOT IN (
       SELECT cluster, count(id) as cnt FROM lf_table
       GROUP BY cluster ORDER BY cnt LIMIT max_clusters )

Algoritm 3: Clusters assignment.
for each (SELECT f.id, f.features FROM lf_table as f) { // all
    SELECT c.id, distance(f.features, features) AS dist
        FROM lf_clusters AS c
        ORDER BY dist LIMIT 1;
    UPDATE lf_table SET cluster=c.id WHERE id=f.id;
}
```

The approach has been tested on a huge problem and a large amount of classes. Performed experiments (to be published) have proven that it is significantly faster than common techniques (linear complexity). The TF-IDF weighting and cosine distance has been then used to accomplish the following task.

3.2. Content-based copy detection pilot

Well, there is nothing more to be written for the task. We have extracted both global and local features, the visual dictionary has been created and search was performed as described in the previous chapters for the run BrnoU.v.fofr1. Some experiments were done about the weighting of the features, however there was no enough time to do this. The “super fast” (“fofr” in Czech) query looked like this (FROM and WHERE clauses are unimportant, qr_tables belong to the transformed shots):

```
SELECT sft.video AS video, sft.frame AS frame,
```

```
rating_cosine(qr_sft.sift, qr_sft.weights, sft.sift, sft.weights) AS sift_rating,
sqrt(distance_square_int4(qr_clr.features, clr.features)) AS color_distance,
sqrt(distance_square_int4(qr_gab.features, gab.features)) AS texture_distance,
sqrt(distance_square_int4(qr_clh.features, clh.features)) AS colhist_distance,
sqrt(distance_square_int4(qr_gra.features, gra.features)) AS grad_distance,
(color_distance - texture_distance - colhist_distance - grad_distance + 777) AS rating
```

These optimized functions, implemented in native C, to be used in the PostgreSQL [16] database are to be published at sourceforge.net. The working name of the project is pgSiftOrder.

3.3. High-level feature extraction

The most important parts of the system in the figure 8 are these:

a) Low-level feature extractors – These are described in section 3.1. Several feature extractors are employed, their feature vectors are concatenated to make a per-frame feature vector. Note that the features are relatively generic. Feature range normalization is part of the feature extraction process.
b) SVM training and cross-validation – Using grid-search, the SVM kernels are optimized and for each high-level feature to be searched, one model is selected by the Model Selection module.
c) Per-frame SVM evaluation – Evaluates the low-level feature vector for each frame in the testing dataset based on the selected SVM model for each high level feature.
d) Per-shot decision – Judges the set of per-frame classifications based on the shot-boundary reference to make a decision on each shot of the testing video dataset.

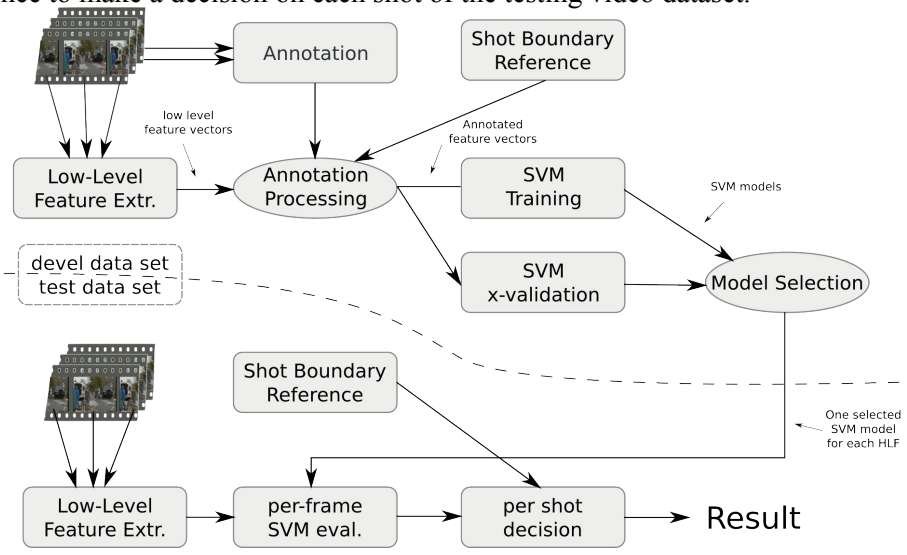


Figure 8. The structure of the high-level feature extraction system.

The system can be described as a brute-force approach to high-level feature extraction, since the low-level features are rather generic than built to match specially the high-level concepts looked for. Also the main classification machine (SVM) is generic. The only specialized part of the system is the per-shot decision making subsystem, which constitutes a very simplistic decision tree.

Composed per-frame feature vector and its processing

All the partial feature vectors mentioned in the previous text are concatenated to make a per-frame feature vector which serves as input to the per-frame classifier described in the next paragraph. This feature vector is normalized across the whole data-set, i.e. maxima and minima for separate features are

found in the training dataset and the features are independently re-scaled so that these maxima and minima correspond to 0.0 or 1.0 respectively. The scaling factors are used for the test data, which can then scaled exceed the normal interval (0-1). These infrequent cases were not found harmful for evaluation by the SVM classifier. These operations were significantly speed up by using database storage, where all features, all videos metadata and embedded re-scalling functions are stored.

Per-frame SVM classifier

LIBSVM [10] was used for classification of separate frames – one classifier was trained for each high level feature to be detected. Given shot annotations were used for all the frames in a given shot. The SVM classifier was trained on 70% of the training data provided, and cross-evaluated on the resting 30%. This cross-evaluation was used for selection of proper parameters of the SVM kernels. The SVM training and parameters selection took majority of the development time (thousands of hours sequentially) and this would be the part most likely to get speeded up in future versions of the system.

Per-shot decision based on per-frame results

For each high level feature separately, the results from per-frame classification are judged for each shot (dropping frames at the beginning and end of each shot to avoid mis-classification). Two quantities are observed in this decision process:

- Positive rate $R = N / P$, where P is the number of positively judged frames and N is number of all frames (excluding the dropped initial and tailing frames in each shot).
- Largest positive sequence s is the length of the longest sequence of positively judged frames.

Two thresholds are defined for these quantities rt, st, exceeding either of them selected the shot being judged for output. These two threshold values were found experimentally on the testing data. The output (positively classified) shots are sorted by the value of r, in cases their number would exceed the given TRECVID limit of positive shots, shots with largest r are selected.

Results, future work

The results of our system for the high-level feature extraction task in TRECVid 2008 were average or slightly below average. We would have expected significantly better results, if we had the time to employ our object classifiers trained for different classes of objects then faces (vehicles, planes, animals), text and local features features entering the joint classification.

On the other hand, the results of such uninformed machine were better than we expected, which could be interpreted in several ways. It surely compliments the selection of low-level features (though the exact set was not optimized in any way), and it shows that in many cases, simple low-level features together with the face detector, mentioned in section 3.1. suffice for a simple base-line solution.

3.4. Search

The scheme of the developed system is illustrated in figure 9. It includes the user (NIST) of the system who provides data and the queries and is a consumer of the results. The video data is being (automatically) annotated in the same manner as in the High-level feature extraction task. So the database is filled with per-keyframe feature vectors including color and texture descriptors, faces found in the image as described in section 3.1. and other data provided by NIST.

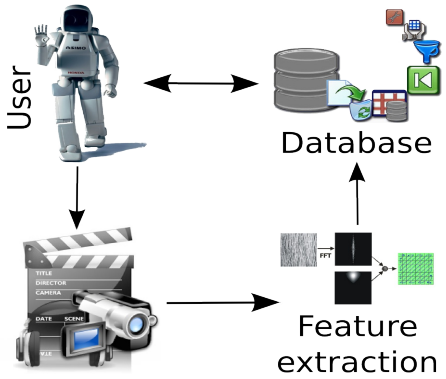


Figure 9. The structure of the multimedia retrieval system.

Text-only runs

We have submitted three runs, first of them was mandatory. These two (F_C_1_BrnoUT_simple_3 and F_C_1_BrnoUT_wordnet_4) were based on the English text of the queries and ASR/MT provided by NIST. The first, classical Information Retrieval (IR, [17]) run F_C_1_BrnoUT_simple_3 used only text tokenizer, Snowball stemmer, TF-IDF weighting, English stop-list were compared by a custom function based on the cosine-distance and used the inverted document index (GIN) for performance reasons [16].

Well, no special research has been made in this area. However its been a lot of practical work, which led into an efficient XML based modeling and visualization [3] together with the efficient native XML database concept, to be published next year.

The run F_C_1_BrnoUT_wordnet_4 moreover used WordNet [11] synonyms and hyponyms of the query keywords to extend the query. We have improved the WordNet to be fast-searchable in the database. However, this resolved in much larger and slower queries. Finally, we didn't expected both runs have exactly the same precision, but we presumed its going to be pretty low.

Video-only runs

The video only run F_C_2_BrnoUT_global_2 is based only on the global image features (color, texture) , aggregated face descriptor, search functions and indexes as described above. The only surprise were pretty good results even there were present no high-level features (except number of faces).

4. Achieved results and the conclusions

Surveillance event detection pilot

In this task, we have submitted PersonRuns, ObjectPut, ElevatorNoEntry and OpposingFlow events, but only PersonRuns and OpposingFlow have evaluate-able results. Nevertheless, the results are incomparable. Because of some technical difficulties (concerning the parallel video processing), we have processed only 20 and 40% of the surveillance videos.

Content-based copy detection pilot

Although the results were little below average, definitely we have the fastest search system. Each query lasted approximately 0.3 seconds, the same as the feature extraction. The speed was achieved using custom C functions for Euclidean and Cosine distance of integer arrays in PostgreSQL database management system. The GIN index has been employed, however we see no extra advantage using it having this amount of data.

The submitted run BrnoU.v.fofr1 was influenced by some technical difficulties – segmentation fault in a C program for visual vocabulary computation caused that approximately half of the shots had only the global features extracted. Not using SIFT for all of the keyframes caused the precision was below average. Thus we presume, next year the results will be much more better.

High-level feature extraction

Our solution can be generally described as a brute-force approach, which relies on generic software pieces (several feature extractors, SVM library, training/evaluation framework for distributed computing), that solve the task in an "uninformed" way. We have sent two results based on the global low-level features and the face detector. The difference between A_Brno_HLF_det_1and A_Brno_HLF_det_2 is that the first run used only 70% of the training data. However, the results are surprisingly the same.

For future implementations of HLF extraction for TRECVID or similar evaluations we intend to include more object detectors and similar frame-processing engines to provide specialized and "informed" knowledge to the overall classification process. These will be represented as mid-level features entering the per-frame classifier. Also many speed-up optimizations have been suggested from the undertaken runs, which would enable more experimenting for future implementations.

Search

We have performed two automatic IR experiments based on the English text of the queries and ASR/MT provided by NIST. The first run F_C_1_BrnoUT_simple_3 used only text tokenizer, Snowball stemmer, TF-IDF weighting, English stop-list were compared by a custom function based on the cosine-distance and used the inverted document index for performance reasons. The run F_C_1_BrnoUT_wordnet_4 used only WordNet synonyms and hyponyms of the query keywords to extend the query. The results of the text-only automatic queries were below average, as expected. However we didn't expected both runs have exactly the same precision.

The run F_C_2_BrnoUT_global_2 based only on the global image features (color, texture) and agregated face descriptor performed surprisingly good. Only two queries were below average, eight of them slightly above and ten queries were "touching the box". However for these queries, no existing system performed really well. The performance is also good, even performed on a computer very distant from the database.

4.1. Overall conclusion

First of all, we have to thank all the people in NIST and groups providing data, transformed data,video and shot references, speech, translations, keyframes, annotations, evaluation metrics and all the human and computer power. We think this is the real force of TRECVid, together with the inspiration from and of all the participants and groups for years.

This is very inspiring for us as a group participated the high level feature extraction task second time and first time in the other tasks. Although this “virginity”, we have made a lot of practical work and some real research, that was concerning mainly to finish the tasks (in time). However, this has ended in a lot of experience, pretty good results in the search task and even too fast results in the copy detection one.

This work has been supported by the research grant project GA AVCR 1ET400750408 Rapid prototyping tools for development of HW-accelerated embedded image-and video-processing applications.", by the project EU-6FP-IST, IST-033812-AMIDA, by the research project "Security-Oriented Research in Information Technology" CEZMSMT, MSM0021630528, and by the research project GACR GA201/06/1821 “Algorithms of Image Recognition”.

References

[1] Smeaton, A. F., Over, P., and Kraaij, W. 2006. Evaluation campaigns and TRECVid. In Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval (Santa Barbara, California, USA, October 26 - 27, 2006). MIR '06. ACM Press, New York, NY, 321-330. DOI = http://doi.acm.org/10.1145/1178677.1178722.

[2] Beran, V. Hradiš M. Zemčík P. Herout A and Řezníček I. 2008. Video Summarization at Brno University of Technology. In Proceedings of the TRECVID BBC Rushes Summarization Workshop 2008.

[3] Chmelař P., Hernych R., Kubíček D. 2008. Interactive Visualization of Data-Oriented XML Documents, In: Advances and Innovations in Systems, Computing Sciences and Software Engineering, IEEE, Springer.

[4] Hradis, M., Herout, A., Zemcik, P. 2008. Local Rank Patterns - Novel Features for Rapid Object Detection, In: Proceedings of ICCVG 2008.

[5] Hradis M. 2008. Framework for Research on Detection Classifiers. In: Proceedings of Spring Conference on Computer Graphics, Budmerice, SK, 2008, s. 171-177.

[6] Mlích, J. And Chmelař, P. 2008. Trajectory classification based on Hidden Markov Models, In: Proceedings of 18th International Conference on Computer Graphics and Vision, Moscow. p. 101-105, ISBN 595560112-0.

[7] Aurenhammer F. – Klein R. 2000. Voronoi Diagrams. Handbook of Computational Geometry, 1 ed. North Holland. ISBN 0444825371.

[8] Berkhin P. 2006. A Survey of Clustering Data Mining Techniques. Grouping Multidimensional Data. p 25-71, ISBN 978-3-540-28348-5.

[9] Carmona E.J., Martinez-Cantos J. and Mira J. 2008. A new video segmentation method of moving objects based on blob-level knowledge. Pattern Recognition LettersVolume 29. p 272-285.

[10] Chang C.C. and Lin C.J. 2001. LIBSVM: a library for support vector machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[11] Fellbaum, C. 1998. WordNet: An Electronic Lexical Database. MIT Press. ISBN 978-0-262-06197-1.

[12] International Telecommunication Union. 1992. Information Technology – Digital Compression and Coding of Continuous-tone Still Images – Requirements and Guidelines. http://www.w3.org/Graphics/JPEG/itu-t81.pdf.

[13] Ho Kyung et al. 2001. MPEG-7 Homogeneous Texture Descriptor. ETRI Journal 23: 41-51. DOI = 10.1.1.17.7368.

[14] Mikolajczyk K. et al. 2005. A Comparison of Affine Region Detectors. International Journal of Computer Vision 65, no. 1 p. 43-72.

[15] 1.Bay H. – Tuytelaars T. – Van Gool L. 2006. SURF: Speeded Up Robust Features. Computer Vision – ECCV 2006, p 404-417.

[16] PostgreSQL Global Development Group. 2008. PostgreSQL 8.3 Documentation: GIN Indexes. http://www.postgresql.org/docs/8.3/static/gin.html.

[17] Van Rijsbergen C. J. 1979. Information Retrieval. Butterworth-Heinemann. ISBN 0408709294. http://www.dcs.gla.ac.uk/Keith/Preface.html.

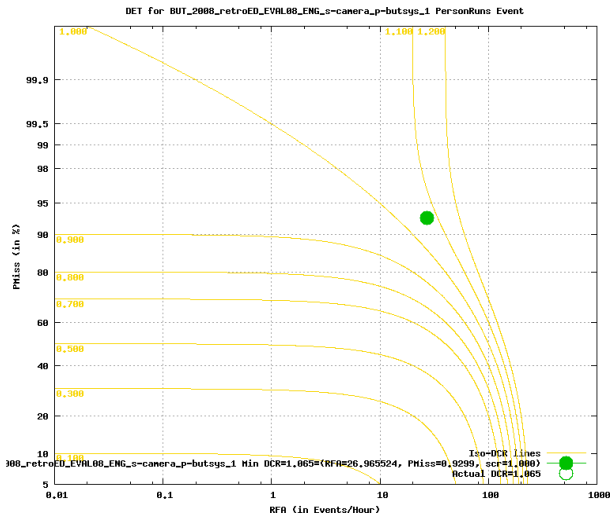
[18] Sivic J. – Zisserman A. 2008. Efficient Visual Search for Objects in Videos. Proceedings of the IEEE 96, no. 4. ISSN 0018-9219.

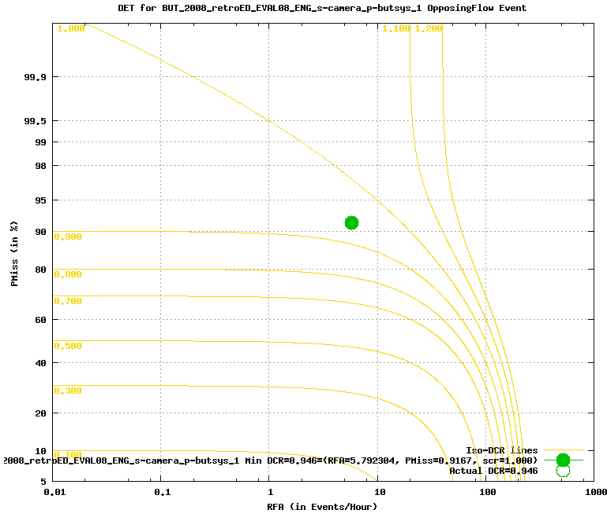
[19] Intel Corporation. 2005. The Open Computer Vision Library. http://opencvlibrary.sourceforge.net/.[FACE2] Sochman, J., Matas, J. 2005. WaldBoost - Learning for Time Constrained Sequential Detection. In: CVPR, (2), 2005, s. 150-156.

[20] Viola, P., Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. In: CVPR.

Appendix A. Surveillance event detection pilot evaluation.

	Corr:YesTarg	Miss:OmitTarg	Miss:NoTarg	FA:YesNontarg	Corr:NoNontarg
ElevatorNoEntry	0	0	0	7	5
ObjectPut	72	1872	0	3366	0
OpposingFlow	1	11	0	296	0
PersonRuns	22	292	0	1378	0
-----	------	------	------	------	------
Total	95	2175	0	5047	5





Appendix B. High-level feature extraction evaluation.

TRECVID 2008: Feature extraction results

Run ID: A_Brno_HLF_det_1
Processing type: Automatic
System training type: A (common devel.data/annotation)
Priority: 1

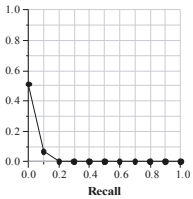
PLEASE NOTE: ALL OF THE MEASURES BELOW ARE BASED ON ASSESSMENT OF A 50% RANDOM SAMPLE OF THE NORMAL SUBMISSION POOLS

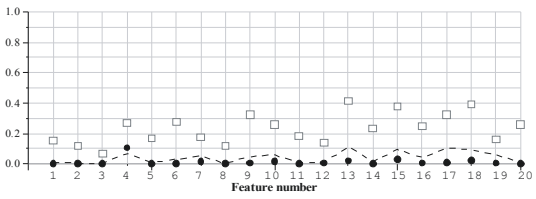
Across 20 test features

Total true shots*: 4670
Total true shots returned*: 272

Mean(inferred average precision)**: 0.012

recall	precision**	n shots	
0.0	0.5090	5	0.2400
0.1	0.0658	10	0.2000
0.2	0.0000	15	0.1734
0.3	0.0000	20	0.1650
0.4	0.0000	30	0.1566
0.5	0.0000	100	0.0860
0.6	0.0000	200	0.0604
0.7	0.0000	500	0.0408
0.8	0.0000	1000	0.0258
0.9	0.0000	2000	0.0136
1.0	0.0000		





* actual counts from a 50% random sample of the normal submission pools
** estimate using 50% sample (e.g., estimated precision = 2 * actual from samp
estimated precision may exceed 1.0

TRECVID 2008: Feature extraction results

Run ID: B_Brno_HLF_det_2
Processing type: Automatic
System training type: B (common devel. data/annotation +)
Priority: 2

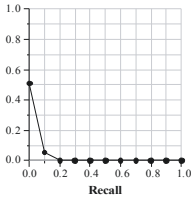
PLEASE NOTE: ALL OF THE MEASURES BELOW ARE BASED ON ASSESSMENT OF A 50% RANDOM SAMPLE OF THE NORMAL SUBMISSION POOLS

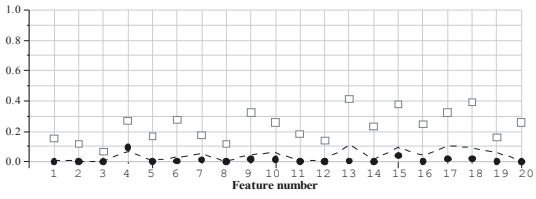
Across 20 test features

Total true shots*: 4670
Total true shots returned*: 236

Mean(inferred average precision)**: 0.012

recall	precision**	n shots	
0.0	0.5096	5	0.2000
0.1	0.0524	10	0.2400
0.2	0.0000	15	0.2000
0.3	0.0000	20	0.1950
0.4	0.0000	30	0.2066
0.5	0.0000	100	0.1030
0.6	0.0000	200	0.0780
0.7	0.0000	500	0.0400
0.8	0.0000	1000	0.0236
0.9	0.0000	2000	0.0118
1.0	0.0000		





* actual counts from a 50% random sample of the normal submission pools
** estimate using 50% sample (e.g., estimated precision = 2 * actual from samp
estimated precision may exceed 1.0

Appendix C. Search and Content-based copy detection pilot evaluation.

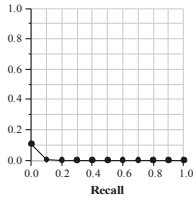
TRECVID 2008: search results

Run ID: BrnoUT_simple
Processing type: automatic
System training type: C (trained other than types A & B)
Condition: 1 (using only video transcript & topic text)
Priority: 3

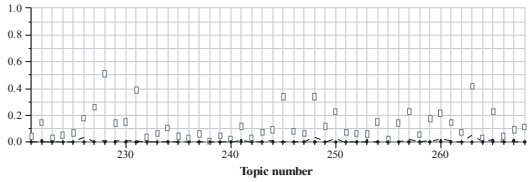
PLEASE NOTE: ALL OF THE MEASURES BELOW ARE BASED ON ASSESSMENT OF A 50% RANDOM SAMPLE OF THE NORMAL SUBMISSION POOLS

Across 48 test topics (221-268)

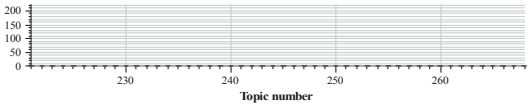
Total relevant shots*: 7333
Total relevant shots returned*: 335
Mean(inferred average precision)**: 0.002



recall	precision**	n shots	
0.0	0.1084	5	0.0256
0.1	0.0026	10	0.0256
0.2	0.0000	15	0.0454
0.3	0.0000	20	0.0468
0.4	0.0000	30	0.0468
0.5	0.0000	100	0.0306
0.6	0.0000	200	0.0254
0.7	0.0000	500	0.0182
0.8	0.0000	1000	0.0142
0.9	0.0000		
1.0	0.0000		



* actual counts from a 50% random sample of the normal submission pools
** estimate using 50% sample (e.g., estimated precision = 2 * actual from samp
estimated precision may exceed 1.0



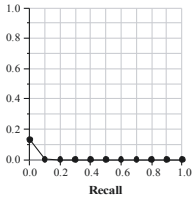
TRECVID 2008: search results

Run ID: BrnoUT_wordnet
Processing type: automatic
System training type: C (trained other than types A & B)
Condition: 1 (using only video transcript & topic text)
Priority: 4

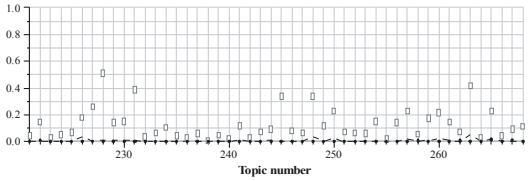
PLEASE NOTE: ALL OF THE MEASURES BELOW ARE BASED ON ASSESSMENT OF A 50% RANDOM SAMPLE OF THE NORMAL SUBMISSION POOLS

Across 48 test topics (221-268)

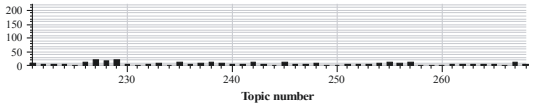
Total relevant shots*: 7333
Total relevant shots returned*: 339
Mean(inferred average precision)**: 0.002



recall	precision**	n shots	
0.0	0.1322	5	0.0334
0.1	0.0008	10	0.0208
0.2	0.0002	15	0.0222
0.3	0.0000	20	0.0334
0.4	0.0000	30	0.0376
0.5	0.0000	100	0.0270
0.6	0.0000	200	0.0204
0.7	0.0000	500	0.0174
0.8	0.0000	1000	0.0142
0.9	0.0000		
1.0	0.0000		



* actual counts from a 50% random sample of the normal submission pools
** estimate using 50% sample (e.g., estimated precision = 2 * actual from samp
estimated precision may exceed 1.0



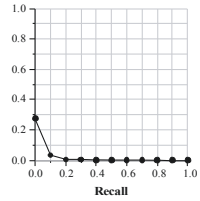
TRECVID 2008: search results

Run ID: BrnoUT_global
Processing type: automatic
System training type: C (trained other than types A & B)
Condition: 2 (as defined by the participant)
Priority: 2

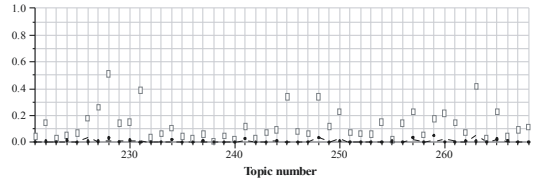
PLEASE NOTE: ALL OF THE MEASURES BELOW ARE BASED ON ASSESSMENT OF A 50% RANDOM SAMPLE OF THE NORMAL SUBMISSION POOLS

Across 48 test topics (221-268)

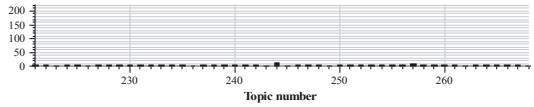
Total relevant shots*: 7333
Total relevant shots returned*: 690
Mean(inferred average precision)**: 0.008



recall	precision**	n shots	
0.0	0.2748	5	0.1304
0.1	0.0344	10	0.1044
0.2	0.0038	15	0.0956
0.3	0.0022	20	0.0934
0.4	0.0000	30	0.0840
0.5	0.0000	100	0.0578
0.6	0.0000	200	0.0464
0.7	0.0000	500	0.0412
0.8	0.0000	1000	0.0300
0.9	0.0000		
1.0	0.0000		



* actual counts from a 50% random sample of the normal submission pools
** estimate using 50% sample (e.g., estimated precision = 2 * actual from samp
estimated precision may exceed 1.0



TRECVID 2008: copy detection results

Run name: BrnoU.v.fofr1
Run type: video-only