University of Marburg at TRECVID 2007: Shot Boundary Detection and High Level Feature Extraction

Markus Mühling[1,2], Ralph Ewerth[1,2], Thilo Stadelmann[1,2], Christian Zöfel[2], Bing Shi[2], and Bernd Freisleben [1,2]

[1] SFB/FK615, University of Siegen, D-57068 Siegen, Germany
[2] Dept. of Math. and Computer Science, University of Marburg, D-35032 Marburg, Germany
{muehling, ewerth, stadelmann, zoefelc, shib, freisleb}@informatik.uni-marburg.de

Abstract

In this paper, we summarize our results for the shot boundary and high level feature detection task at TRECVID 2007. Our shot boundary detection approach of previous TRECVID evaluations served as a basis for our experiments this year and was modified in several ways. First, we have incorporated a new metric selection for cut detection based on the evaluation of a clustering result. Second, we have tested the possibility to improve cut detection results via self-supervised learning. Third, the unsupervised approach for gradual transition detection has been supplemented with a false alarm removal method using a state-of-the art camera motion estimation approach. Regarding high-level feature detection, one focus of this year's task was to investigate the question how well a trained system generalizes from the TRECVID 2005 news data to this year's Sound and Vision data. However, only two institutes have submitted four runs of the related type "a" for evaluation (three of them were submitted by us). In this paper, we present our experiments for the high-level feature task with respect to the generalization capabilities of our system trained on broadcast news videos. For this purpose, we have conducted several experiments using our system which is based on low-level features as well as on state-of-the-art approaches for camera motion estimation, text detection, face detection and audio segmentation.

1. Structured Abstract

In this section, the results of our participation in both tasks are presented in form of the requested structured abstract. The shot boundary detection approach and the related experimental results are presented in section 2. Our system developed for high-level feature extraction is described in section 3 along with the experimental results. Section 4 concludes the paper.

The following definitions are used in this paper:

$$recall = \frac{\#correctDetectedItems}{\#Items} \quad (1)$$

$$precision = \frac{\#correctDetectedItems}{\#correctDetectedItems + \#falseAlarms} \quad (2)$$

$$f1 = \frac{2 * recall * precision}{recall + precision} \quad (3)$$

The high-level feature detection experiments were evaluated by the TRECVID team using the inferred average precision measure suggested in [22].

Shot Boundary Detection: "What approach or combination of approaches did you test in each of your submitted runs?"

The investigated unsupervised approach relies on our TRECVID system of previous years [6, 8]. This approach does not need any training data and utilizes k-means clustering to achieve robust results for both cut detection and gradual transition detection.
To detect cuts, two different frame dissimilarity measures are applied: Motion-compensated pixel differences and histogram dissimilarities of subsequent DC-frames. A sliding window technique is used to measure the relative height of a peak value in a temporal neighborhood. The best sliding window size is estimated by evaluating the clustering quality of "cut clusters" for several window sizes [4]. In addition to our previous system, clustering quality is also used to select the most appropriate dissimilarity metric for a particular video. Furthermore, the unsupervised cut

detection approach is extended by self-supervised learning [7] in order to obtain a robust result on a particular video.
To detect gradual transitions, a temporal multi-resolution approach is applied and dissimilarities are measured for several frame distances. Feature vectors are created similar to the cut detection approach using a sliding window technique. K-means is applied to cluster these feature vectors. This approach is extended by a fade detector following the proposal in [20]. In addition to our last year's system, false alarms caused by camera motion are removed by employing the results of our state-of-the-art camera motion estimation approach [5].
Nine runs were submitted, and the different parameter settings for each run are described in Table 1.

Shot Boundary Detection:
"What, if any significant differences (in terms of what measures) did you find among the runs?"
"Based on the results, can you estimate the relative contribution of each component of your system/approach to its effectiveness?"

In terms of the f1-measure, the best run for cut detection was achieved using the self-supervised learning approach (f1=0.946). In particular, this run achieved the best precision among all submitted runs. Regarding false alarm removal for gradual transitions, the incorporation of the camera motion approach improved the results significantly. Interestingly, the consideration of pan detection (f1: 0.66-0.67) was more efficient than considering the detection of all motion types (f1: 0.55-0.56). Nonetheless, the consideration of all motion types still yielded better results than the baseline system without false alarm removal (f1: 0.35-0.44).

Shot Boundary Detection: "Overall, what did you learn about runs/approaches and the research question(s) that motivated them?"

The cut detection approach is rather robust, the self-supervised learning approach improved the results only slightly (in case when enough features are used), mostly in terms of precision.
The incorporation of camera motion estimation was very useful in order to reduce the number of false alarms for gradual transition detection. It turned out that detections of horizontal camera motion (pan) were more suited for false alarm removal than considering all motion types (pan, tilt and zoom).

High-level Feature Extraction: "What approach or combination of approaches did you test in each of your submitted runs?"

The following runs were submitted:

- a_ma1: Baseline, TRECVID 2005 training set with ground truth from Mediamill challenge system [19]
- A_ma2: Baseline07, TRECVID 2007 training set, merged annotations from active learning and MCQ-ECT-CAS
- A_ma3: Baseline07 plus additional distinction between color and gray-scale shots
- A_ma4: A_ma7 plus additional distinction between color and gray-scale shots
- a_ma5: Context features for 101 Mediamill concepts, training set of Mediamill challenge system (subset of TRECVID 2005 training set)
- a_ma6: a_ma5_6 plus transductive learning
- A_ma7: Additional evaluation, context features for 101 Mediamill concepts, TRECVID 2005 and 2007 training set

High-Level Feature Extraction: "What, if any significant differences (in terms of what measures) did you find among the runs?"

To investigate the generalization capabilities of our system trained on broadcast news videos, we performed three experiments of the related category "a". First, we compared our baseline system by using either the TRECVID 2005 development set or the TRECVID 2007 development set for system training (a_ma1 respectively A_ma2). In a second experiment (a_ma5), the generalization capabilities of context features have been investigated and finally, in the third experiment (a_ma6), we applied transductive learning to the set of context features, realized by transductive support vector machines [11]. The idea concerning the transductive learning approach is to improve the detection performance of those high-level features whose appearance is strongly related to the Sound and Vision video data. Further experiments of category "A" investigated the usefulness of distinguishing color and gray-scale images and consequently separate training data for each modality as well as the impact of context features.

High-Level Feature Extraction: "Based on the results, can you estimate the relative contribution of each component of your system/approach to its effectiveness?"

Even though the baseline system trained on the Sound and Vision data yields a clearly better result in terms of mean inferred average precision (7.03% compared to 4.94%), several of the high-level feature models trained on the broadcast news videos generalize very well to the Sound and Vision video data (e.g. “car”, “charts”, “airplane”, “military” and “sports”). The second run related to category “a” investigating the generalization capabilities of context features showed a slight performance decrease compared to the baseline system (4.94% to 4.55% mean inferred average precision). However, the high-level features “sports” and “people marching” benefited from using context vectors. Interestingly the use of context concerning category “a” (a_ma5) outperformed the corresponding category “A” system (A_ma7) in terms of mean inferred average precision (4.55% vs. 4.08%).
No performance gain could be obtained by transductive learning compared to the reference system (3.31% vs. 4.55% mean inferred average precision). In comparison to the reference system, the retrieval results could be improved slightly for only a few high-level features, but these improvements might not be significant. Despite that, this approach achieved our best results for the high-level features “flag-us” and “airplane”.
The distinction between color and gray-scale images (keyframes) and consequently learning different models for each modality ended up with similar performances for nearly all high-level features, showing slight reductions in comparison to the reference systems (7.03% vs. 6.69% and 4.08% vs. 3.87% mean inferred average precision).

High-Level Feature Extraction: “Overall, what did you learn about runs/approaches and the research question(s) that motivated them?”

The experiments revealed that the generalization capabilities of systems trained on broadcast news videos to the sound and vision data are limited. However, several high-level feature models generalized very well to the sound and vision data like “car”, “explosion_fire”, “airplane”, “mountain” or “sports”.
The experiments also showed that the use of context features brought no performance gain and the impact of distinguishing between color and gray-scale images had nearly no impact on the detection results of the high-level features.

2. Shot Boundary Detection

The shot boundary detection approach is split up into two parts in order to detect cuts and gradual transitions appropriately. It is basically the same approach as in TRECVID 2006 except for the modifications described in the next three subsections.

2.1 Automatic Metric Selection

In our baseline system for cut detection, the best sliding window size is estimated by evaluating the clustering quality of the “cut” clusters for several window sizes [4]. In addition, we employ the clustering quality also to select the most appropriate dissimilarity metric for a particular video. The clustering results of histogram-based dissimilarity values and motion compensated pixel differences are compared with respect to the quality of the resulting “cuts” cluster. It is evaluated using the Silhouette coefficient [4].

2.2 Self-Supervised Learning for Video Cut Detection

Last year, we have extended the unsupervised basic system with several unsupervised classifiers. This year, we investigated the possibility to extend the basis system with two additional supervised classifiers which are trained directly (without any supervision) and automatically on the test video under consideration.
It has been shown that an ensemble of classifiers can improve accuracy in recognition tasks [13]. Since most transitions in a video are abrupt (without any transitional frames between the different shots), self-supervised learning is applied for cut detection. The extension of the basis system to an ensemble with self-supervised classifiers works as follows. First, cuts are detected in a video using the basis system. The detection results are employed as training data for the subsequent processing steps. Then, the best features are selected for the given video via Adaboost (as described in [21]). For cut detection, we have defined 42 features for two frame distances (1 and 2) describing the dissimilarity of DC-frames with respect to:

- motion compensated pixel differences,
- histogram differences,
- luminance mean and variance,
- edge histograms of Sobel-filtered (vertically and horizontally) DC-frames,
- local histogram differences, and
- ratio of the second largest dissimilarity value divided by the local maximum for several sliding window sizes.

The selected features are split up into two disjoint sets and two SVMs are trained (using only the automatically labeled data of the particular test video) on either feature set. Thus, we finally get three classifiers evaluating each frame (considering the unsupervised approach as a kind of classifier as well). A majority vote is employed, i.e. a cut is detected if at least two "experts" vote that a frame belongs to a new shot. Details of this approach can be found in [7].

2.3 Gradual Transition Detection

The proposed approach for gradual transition detection consists of three main components: a fade detector, an unsupervised gradual transition detector, and false alarm removal based on camera motion estimation. The general gradual transition detection process is preceded by fade detection [20]. The main idea of the proposed approach for gradual transition detection is to view a gradual shot change as an abrupt shot change at a lower temporal resolution. For this purpose, subsampled frame dissimilarity time series are used. Given a time series with a frame distance m which is subsampled accordingly by factor m, a gradual transition of length $n \leq m/2$ should be represented by an isolated peak in the time series - as it is the case for a cut at the highest temporal resolution. Feature vector creation and the clustering process take place in a similar way as in the cut detection approach. Finally, false alarms are removed based on the results of a high-quality motion estimation algorithm [5]. Summarizing, the main components of the unsupervised gradual transition detection are:

1. Fade detection according to [20];
2. Unsupervised gradual transition detection
    a. Measure frame dissimilarities for several frame distances and create feature vectors;
    b. Cluster feature vectors for several frame distances in one or several clustering processes;
    c. Post-processing of cluster(s) – there might be several clustering results for the different frame distances, i.e. results must be united appropriately;
3. False alarm removal, based also on camera motion estimation.

A gradual transition candidate is removed in case it is completely covered by a camera motion event (either pan or of any type). The start and end positions of the remaining transitions are refined by comparing the histogram dissimilarity.

Details of our approach can be found in [8].

2.4 Experimental Results

The MDC decoder was used for MPEG decoding [14] in our experiments. The shot detection approach was tested with the following parameter settings for all runs. The baseline system with one metric uses 1D-histogram (YUV color space, with 512 bins) dissimilarities of DC-frames, the "2metrics" runs use also motion compensated pixel differences of DC-frames. The range of possible sliding window sizes was between 5 and 20. In case of automatic metric selection, motion compensated pixel differences are used as well. The frame distances for gradual transition detection were set to: 6, 10, 20, 30, 40, 50; the parameter describing the initial sliding window size for the finest temporal resolution was set to 24. The experimental settings and the results for the different runs are shown in Table 2 and 3.

Overall, shot boundary detection worked very well: Regarding all transitions, our best run achieved an f1-measure of 0.92. The cut detection performance was very good: recall is about 95% and precision about 93% in nearly all runs, yielding an f1-measure of about 0.94. The results did not differ with respect to automatic metric selection. For self-supervised learning, using only 11 features was not sufficient and performance degraded. However, our best result for cut detection in terms of the f1-measure (0.946) was obtained for the self-supervised learning approach using 45 features, in particular, precision was improved and recall decreased only slightly. In this run (marburg-self45), gradual transition detection has been disabled to measure only the performance of the self-supervised cut detection approach.

For gradual transition detection, the results vary noticeably depending on the used false alarm removal. The incorporation of camera motion estimation improved the results for gradual transition detection significantly (f1 of best run: 0.67). Only three institutes obtained better results for gradual transition detection. When horizontal camera motion (pan) was employed, precision was clearly increased by nearly 30%, while recall decreased only very slightly by 1.6% (both compared to the baseline system). In this best case, the highest precision (57.0% and 59.5%) could be achieved for our runs while recall was nearly as high as without false alarm removal (about 77%).

Run	Cuts: Metric Selection	Cuts: Self-Supervised Learning (#Features)	Graduals: False Alarm Removal
marburg-2metrics-motion-all	Yes	No	All Motion Types
marburg-2metrics-motion-pan	Yes	No	Pan
marburg-2metrics	Yes	No	No
marburg-base	No	No	No
marburg-motion-all	No	No	All Motion Types
marburg-motion-pan	No	No	Pan
marburg-self11-motion-pan	Yes	Yes (11)	Pan
marburg-self11	Yes	Yes (11)	No
marburg-self45	Yes	Yes (45)	-

Table 1: The parameter settings for the different runs: The columns indicate whether metric selection or self-supervised learning (both for cuts) was applied, and whether and how a false alarm removal based on camera motion took place for gradual transition detection.

Run	Cuts		Gradual Transitions		Gradual Trans. Frame-based		All Transitions	
	Recall	**Prec.**	**Recall**	**Prec.**	**Recall**	**Prec.**	**Recall**	**Prec.**
marburg-2metrics-motion-all	95.5%	93.3%	53.7%	57.7%	*46.7%*	*90.4%*	92.2%	90.7%
marburg-2metrics-motion-pan	95.7%	93.0%	77.7%	57.0%	43.7%	90.6%	94.2%	89.3%
marburg-2metrics	96.0%	91.5%	79.3%	28.3%	43.7%	90.2%	94.6%	79.5%
marburg-base	94.6%	93.0%	78.2%	30.6%	43.9%	90.1%	93.3%	81.7%
marburg-motion-all	94.4%	94.4%	53.2%	61.0%	46.6%	90.2%	91.1%	92.1%
marburg-motion-pan	94.5%	94.2%	*76.6%*	*59.5%*	44.0%	90.6%	*93.1%*	*90.7%*
marburg-self11-motion-pan	81.4%	93.0%	40.4%	43.7%	47.8%	89.7%	78.1%	88.8%
marburg-self11	90.6%	91.9%	73.9%	22.9%	44.5%	89.4%	89.3%	76.5%
marburg-self45	*93.2%*	*96.0%*	-	-	-	-	85.7%	96.0%

Table 2: Recall and precision for the different runs, separated for cuts, gradual transitions, for gradual transitions on a frame basis, and for all transitions.

Run	Cuts	Gradual Transitions	Gradual Transitions Frame-based	All Transitions
marburg-2metrics-motion-all	0.944	0.556	**0.616**	0.914
marburg-2metrics-motion-pan	0.943	0.658	0.590	0.917
marburg-2metrics	0.937	0.417	0.589	0.864
marburg-base	0.938	0.440	0.590	0.871
marburg-motion-all	0.944	0.568	0.615	0.916
marburg-motion-pan	0.944	**0.670**	0.592	**0.919**
marburg-self11-motion-pan	0.868	0.420	0.624	0.831
marburg-self11	0.913	0.350	0.594	0.824
marburg-self45	**0.946**	-	-	0.906

Table 3: F1-measures for all runs, separated for cuts, gradual transitions, frame-based detection performance of gradual transitions, and all transitions.

3. High-level Feature Extraction Task

In this section, we present our system for high-level feature extraction. First, in section 3.1 we describe the automatically extracted low-level features plus additional mid-level features, which are the result of state-of-the-art algorithms in the field of camera motion estimation [5], text detection [9], face detection [21] and audio segmentation. The following parts of our system are discussed in detail in sections 3.2 and 3.3. Finally, the experimental results are presented in section 3.4.

3.1 Low-Level and Mid-Level Features

Our video analysis system automatically extracts several low-level as well as mid-level features. In section 3.1.1 we describe our visual features, followed by the audio features in section 3.1.2. Furthermore, we created context information based on 101 high-level concepts. These context features are described in section 3.1.3.

3.1.1 Visual Features

Several visual features are extracted for each video shot. The frame in the middle of a shot is used as keyframe. If a keyframe contains black bars, these top and bottom regions of the image are automatically detected and removed in a preprocessing step. The removal of the black bars is realized by zooming into the image. In addition to the keyframe based low-level features (color moments, color correlograms, texture and gabor features) several mid-level features are extracted automatically from the entire shot by utilizing camera motion estimation [5], face detection [21] and text detection [9]. In the following, the extracted features are briefly described.

Color moments: Color moments are extracted at two different granularities. The first three global color moments are computed for the whole keyframe. Corresponding values are extracted for each region of a 3 x 3 grid in HSV (Hue, Saturation, Value) color space. The i-th pixel of the j-th color channel of an image region is represented by c_{ij}. Then, the first three color moments are defined as:

$$mean_j = \frac{1}{N} \cdot \sum_{i=0}^{N-1} c_{ij} \tag{4}$$

$$stdev_j = \sqrt{\frac{1}{N} \cdot \sum_{i=0}^{N-1} (c_{ij} - mean_j)^2} \tag{5}$$

$$skew_j = \sqrt[3]{\frac{1}{N} \cdot \sum_{i=0}^{N-1} (c_{ij} - mean_j)^3} \tag{6}$$

Texture features: The gray-scale image co-occurrence matrices m_k are constructed at 8 orientations. We use these matrices to extract the following values representing the global texture:

$$energy_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (m_{kij})^2 \tag{7}$$

$$contrast_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (i-j)^2 \cdot m_{kij} \tag{8}$$

$$entropy_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} m_{kij} \cdot log(m_{kij}) \tag{9}$$

$$homogeneity_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \frac{m_{kij}}{1+|i-j|}, \tag{10}$$

where N is the number of gray values and m_{kij} is the value of the co-occurrence matrix m_k at position (i, j).

Color autocorrelograms: Color correlograms describe the spatial relationship between colors, whereas auto-correlograms are limited to identical colors. An autocorrelogram expresses the probabilities of colors re-occurring in a certain distance. We preferred small distances (1, 4, 7, 10, 13, 16 and 19 pixel), so that local spatial correlations of identical colors are represented by the correlogram. Colors are described in HSV (hue, saturation, value) color space. By choosing a smaller number of bins representing the brightness component we get more independent of illumination changes. In total, each color correlogram results in a 350-dimensional feature vector.

Gabor wavelet features: Gabor wavelet features are extracted for eight orientations and five frequencies. The functions to compute the wavelet coefficients can be expressed as follows [12]:

$$g_{\theta,\lambda,\varphi,\sigma,\gamma}(x,y) = e^{-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}} \cos(2\pi \frac{x'}{\lambda} + \varphi)$$
$$x' = x\cos\theta + y\cos\theta \qquad (11)$$
$$y' = -x\sin\theta + y\cos\theta$$

A Gabor wavelet is controlled by five parameters: orientation θ, wave length λ, phase φ, radius σ of the Gaussian function, and the aspect ratio γ. The radius of the Gaussian function is chosen proportionally to the wave length, and the aspect ratio is fixed to 1. Gabor energies of a pixel for the different orientation and spatial-frequency combinations are obtained by a superposition of the phases 0 and π/2 using the L2-Norm. The resulting 40 Gabor energies per pixel are summarized in a Gabor histogram describing the whole image. By distinguishing ten energy classes we obtain 400 histogram feature values. We further compute the average result of each Gabor energy filter for each region of a 4 x 4 grid. Thus, the total number of Gabor wavelet features amounts to 1040 values.

Camera motion features: Motion vectors embedded in MPEG videos are employed to compute camera motion at the granularity of P-frames, according to the approach presented in [5]. The following camera motion types are distinguished: translation along the x-axis, respectively y-axis, rotation around the x-axis, respectively y-axis and z-axis, and zoom. The distribution of the values for a shot concerning the different camera motion types are described by using the following statistical values: mean, median, minimum, maximum, standard deviation, and skewness. Additionally, the percentages of a shot concerning the different camera motion types pan, tilt and zoom are considered, so that we finally get a 39-dimensional camera motion vector.

Text features: A robust text detection approach [9], which can automatically detect horizontally aligned text with different sizes, fonts, colors and languages, is applied at the granularity of I-frames. First, a wavelet transformation is applied to the image and the distribution of high-frequency wavelet coefficients is considered to statistically characterize text and non-text areas. Then, the k-means algorithm is used to classify text areas in the image. The detected text areas undergo a projection analysis in order to refine their localization. We use the detected text areas to derive the following features per shot: the number of appearing text elements, the average text position, the mean text frame coverage, and the average number of text elements per frame.

Face features: Frontal and profile faces are detected in each video frame using the face detector provided by Intel's OpenCV library [www.intel.com/technology/computing/opencv]. The face detection approach is an implementation of the approach suggested by Viola and Jones [21] with Lienhart's extensions [15]. The Adaboost-based approach of Viola and Jones was chosen since it is a very fast approach that nearly operates in real-time on today's computers and thus can even be applied to every single frame of a sequence. Since their approach usually reports many detections of slightly different sizes and positions, an average rectangle is computed based on the reported detections, in case that the number of detections exceeds a threshold. A tracking procedure also based on the OpenCV library is used to assemble face appearances of the same person in subsequent frames of a shot using the optical flow computation of Bouguet [2], which is an extension of the Lukas-Kanade [18] algorithm. The extension processes image pyramids to enable the estimation of fast movements as well. For each shot, the number of face sequences, the number of detected faces, front faces respectively profile faces, the average frontal respectively profile shot size, the mean number of detection hits for frontal faces respectively profile faces and the percentage length of a shot, where a person appears, are considered as mid-level features.

3.1.1 Audio Features

For analyzing audio data, we extracted several low-level audio features, which are fed into a content-based audio classification and segmentation system based on the approach of Lu et al. [17]. The following low-level features are extracted from non-overlapping 25ms frames [17]: 8th-order mel frequency cepstrum coefficients, zero crossing rate, short time energy, sub-band energy distribution, brightness and bandwidth, spectrum flux, band periodicity and a measure of frame noisiness.

The audio classification system produces mid-level features on a per-second (sub-clip) basis in the form of acoustic class labels and related probabilities for "silence", "speech", "pure speech", "non-pure speech", "music", "background" and "action" sounds (an error label, "undefined", may also be produced). The low-level features are therefore aggregated per second, normalized and then concatenated to form one feature vector per sub-clip, which is processed by a hierarchical tree of support vector machines, if it was not previously classified as silence by a threshold based classifier. Figure 1 shows this classification tree, which is trained on more than 32 hours of audio samples including, among others, the TIMIT data for clean speech [16] and the NOIZEUS [10] corpus. Five-fold cross-validation on a subset of 15000 feature vectors was used to find the best parameter settings for each two-class support vector machine with a RBF (radial basis function) kernel using the libSVM library [3]. Finalizing the classifier's decision, short silence periods within speech are labeled as "pause" by a heuristic decision function. A second algorithm based on the work of Ahmadi and Spanias [1] processes energy, zero-crossing rate and cepstral peak low-level features to add "voiced" and "unvoiced" speech labels to the mid-level features.

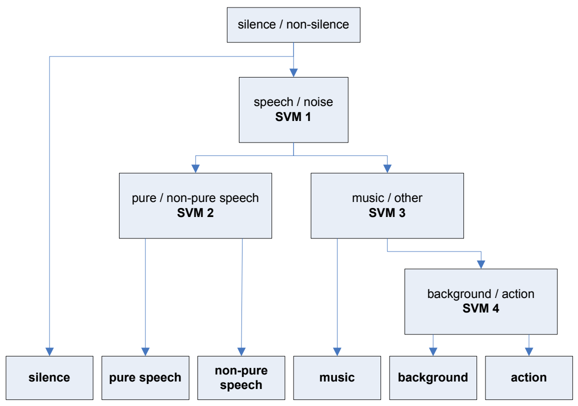

Figure 1: Scheme of the hierarchical audio type classifier: A single feature vector per sub-clip serves as input; output is a single acoustic class label and its corresponding probability.

All 11 mid-level features are then processed to describe the audio-content of a video shot by statistical values: mean, median, minimum, maximum, standard deviation, and skewness of the per-frame label-probabilities are calculated. Furthermore, the percentage of each audio type label with respect to the shot length is calculated. Finally, these percentages and the distribution properties of the probabilities are fed into the further learning algorithm as the final audio mid-level features, resulting in a 77-dimensional feature vector.

3.1.2 Context Features

Besides the previously mentioned features, context information between high-level concepts were considered. Based on the TRECVID 2005 training set 101 concept models related to the Mediamill annotations [19] were built using support vector machines, which is described in more detail in the next section. The probability output of the support vector machines form a 101-dimensional model vector, which describes the context information of a shot.

3.2 High-Level Feature Detection System

The goal of the proposed system is to learn models for the high-level semantic features based on the extracted audiovisual low-level and mid-level features described in section 3.1. In our baseline system, we concatenated the multi-modal low-level and mid-level features in an early fusion scheme and fed them directly into a support vector machine with a radial basis function kernel using the implementation provided by the libSVM library [3].
We also extended our system to consider context information. Here, the model vectors described in section 3.1.2 serve as input for a further support vector machine, which learns the context between the 101 Mediamill concepts for each high-level feature.

3.2.1 Distinguishing Color and Gray-Scale Images

Several of the 36 high-level features like "sky", "vegetation", "flag-us", "snow", "road" or "waterscape" seem to be strongly related to color features. We randomly selected 2099 color and 1210 gray-scale images (key frames) from the sound and vision training set. The colors of an image are transformed to the HSV (hue, saturation, value) color space and the both image subbands hue and saturation are described by the statistical values mean, standard deviation, median and skewness. Together with two 20-dimensional histograms based on the hue respectively saturation subband of the image we formed a 48-dimensional feature vector (the brightness values are ignored). A support vector machine with a RBF kernel provided by [3] is used to build a model based on these feature vectors, which is able to distinguish between color and grey-scale images. This subdivision of the sound and vision training set is exploited to build separate high-level feature models for each modality. The decision, which model should be applied to a shot in the sound and vision test set, is based on the prediction of the previously described support vector machine.

3.2.2 Transductive Learning

The appearance of certain high-level features is strongly related to contextual information. For example, the appearance of semantic concepts, such as e.g. entertainment or news anchors, is determined by the used editing layout which usually is typical for a certain broadcasting station. The idea of our transductive learning approach is to adapt the appearance models of the semantic concepts based on broadcast news videos to the Sound and Vision video data and thus avoiding the time consuming task of creating a training set for the new kind of video data. In a transductive setting, in addition to the training samples, the unlabeled test samples are considered in the learning process as well. We applied transductive learning to the set of context features, realized by transductive support vector machines [11].

3.3 Experimental Results

In this section, we present our results for the high-level feature detection task. We submitted three runs for each of the related categories "a" and "A" plus an additionally evaluated category "A" run. The focus was on the question how well a trained system generalizes from the TRECVID 2005 news data to this year's Sound and Vision data. First, we compared our baseline system by using either the TRECVID 2005 development set or the TRECVID 2007 development set for system training. Even though the system trained on the Sound and Vision data yields a clearly better result in terms of mean inferred average precision (7.03% compared to 4.94%), several of the high-level feature models seem to generalize very well to the Sound and Vision video data, e.g. "car", "charts", "airplane", "military" and "sports" (see Figure 3).
In a second experiment (a_ma5), the generalization capabilities of context features were investigated. Due to lack of time, we limited the training samples to the training set of the Mediamill challenge, which is a subset of the TRECVID 2005 training set. Overall, this system showed a slight performance decrease compared to the baseline system (4.94% to 4.55% mean inferred average precision). However, the high-level features "sports" and "people marching" benefited from using context vectors.
In the third experiment, we applied transductive learning to the set of context features, realized by transductive support vector machines. The idea was to improve the detection performance of those high-level features whose appearance is strongly related to the Sound and Vision video data. Overall, no performance gain could be obtained by transductive learning in terms of mean inferred average precision compared to the reference system (3.31% vs. 4.55%). In comparison to the reference system (a_ma5), the retrieval results could be improved slightly for only a few high-level features, but these improvements might not be significant. Despite that, this approach achieved our best results for the high-level features "flag-us" and "airplane".

We conducted two further experiments related to category "A" investigating the impact of distinguishing color and gray-scale images and context features. Classifying gray-scale images was more difficult than expected, because several of the video shots show old discolored gray-scale images or videos (many of the keyframes indicate a red cast). Furthermore, dark colors in gray-scale images do not necessarily have small hue and saturation values. Finally, the distinction between color and gray-scale images and consequently learning different models for each modality ended up with similar performances for nearly all high-level features in comparison to the reference systems, showing slight reductions in terms of mean inferred average precision (7.03% vs. 6.69% and 4.08% vs. 3.87%).

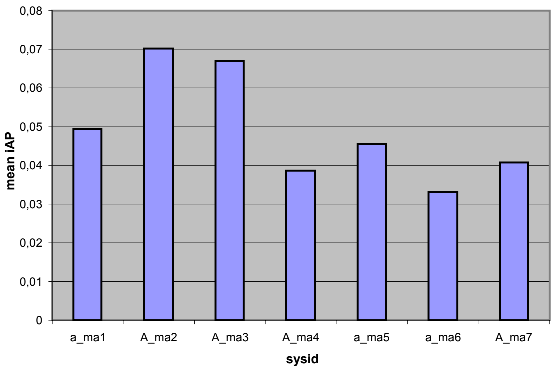

Figure 2: Overview of the results of our six runs in terms of mean inferred average precision.

Furthermore, we investigated the impact of context features based on the 101 concept models related to the Mediamill annotations. The performance of these context systems (A_ma4 and A_ma7) suffers from heavy performance losses (from 7.03%/6.69% to 4.08%/3.87% mean inferred average precision). Interestingly, the use of context features concerning category "a" shows only a slight decrease (from 4.94% to 4.55% mean inferred average precision) and thus outperforms the category "A" systems. Further investigations are necessary to find the reasons for this performance loss.
Overall, the best result in terms of mean inferred average precision was achieved by our baseline system using the sound and vision training set (7.03% mean inferred average precision).

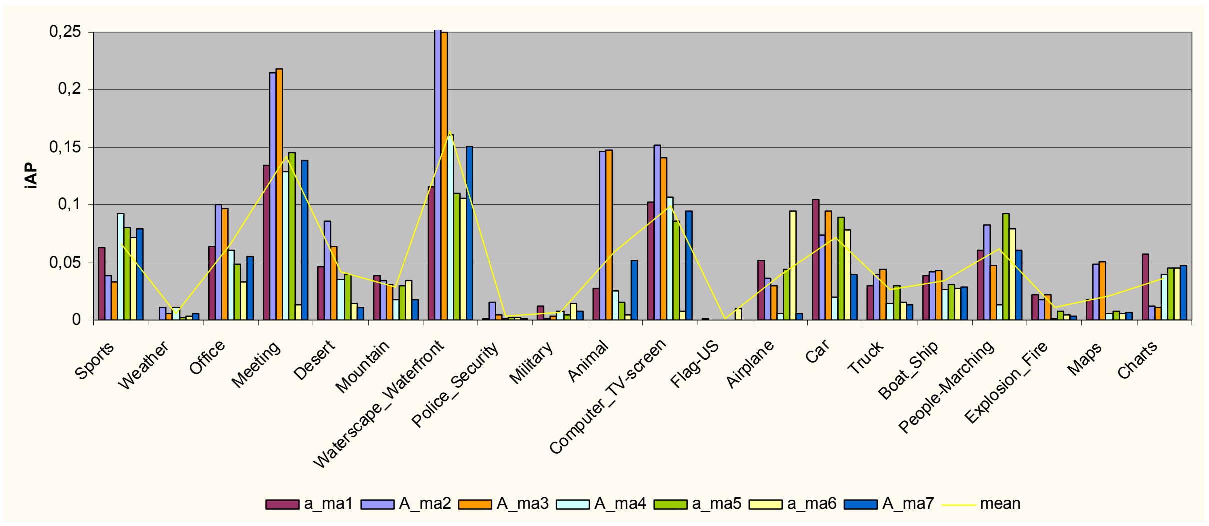

Figure 3: Comparison of our submitted runs plus an additional one concerning all evaluated high-level features. The mean values refer to the average of all our runs related to the categories "a" and "A".

4. Conclusions

In this paper, we have presented our experiments for two tasks at TRECVID 2007: shot boundary detection and the high-level feature detection task.
Our shot boundary detection system of last year was extended in several ways. First, we have incorporated an automatic metric selection for cut detection based on the evaluation of a clustering result. Second, we have tested the possibility to improve cut detection results via self-supervised learning. Third, the unsupervised approach for gradual transition detection has been supplemented with a false alarm removal using a state-of-the art camera motion estimation approach.
Overall, shot boundary detection worked very well: Regarding all transitions, our best run achieved a f1-measure of 0.92. The cut detection performance was very good, achieving a f1-measure of about 0.94 in nearly all runs. The best run based on self-supervised learning obtained a f1-measure of 0.946. The incorporation of camera motion estimation improved the results for gradual transition detection significantly (f1 of best run: 0.67). In case when horizontal camera motion was employed, precision clearly increased by nearly 30%, while recall decreased only very slightly by 1.6% (both compared to our baseline system).
The experiments for high-level feature detection revealed that the generalization capabilities of systems trained on broadcast news videos to the sound and vision data are limited. Systems trained on the Sound and Vision data except the context system achieved clearly better results in terms of mean inferred average precision. Our best system achieved a mean inferred average precision of 7.03%. However, several high-level feature models generalize very well to the Sound and Vision video data (e.g. "car", "charts", "airplane", "military" and "sports").
The experiments also showed that the impact of distinguishing between color and gray-scale images is negligible, it had nearly no impact on the detection results of single high-level features. It is planned to further investigate the influence of the gray-scale image detection performance on the final result.
In total, we achieved the second best result for the high-level features "meeting" and "people marching", probably due to our face processing approach. For 10 out of the 20 evaluated high-level features, we were among the top seven teams.

5. Acknowledgements

This work is financially supported by the Deutsche Forschungsgemeinschaft (SFB/FK 615, Teilprojekt MT).

6. References

1. Ahmadi, S. and Spanias, A. S. Cepstrum-Based Pitch Detection Using a New Statisctical V/UV Classification Algorithm, *IEEE Trans. on Speech and Audio*, Vol. 7, No. 3, May 1999, 333-338.

2. Bouguet, J.-Y. Pyramidal Implementation of the Lucas Kanade Feature Tracker. In *OpenCV Documentation, Intel Corporation, Microprocessor Labs*, 1999.

3. Chang, C.-C. and Lin, C.-J. LIBSVM: A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm

4. Ewerth, R. and Freisleben, B. Video Cut Detection without Thresholds., *Proc. of 11th Int'l Workshop on Systems, Signals and Image Processing*, Poznan, Poland, 2004, 227-230.

5. Ewerth, R., Schwalb, M., Tessmann, P., and Freisleben, B.: Estimation of Arbitrary Camera Motion in MPEG Videos. In *Proc. of the 17th International Conference on Pattern Recognition*, Vol. 1. Cambridge, United Kingdom, 2004, 512-515.

6. Ewerth, R., Beringer, C., Kopp, T., Niebergall, M., Stadelmann, T., and Freisleben, B. University of Marburg at TRECVID 2005: Shot Boundary Detection and Camera Motion Estimation Results. In *Online Proceedings of TRECVID Conference Series 2005*: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html

7. Ewerth, R. and Freisleben, B.: Self-Supervised Learning for Robust Video Indexing. In *Proceedings of the IEEE International Conference on Multimedia & Expo*, 2006, Toronto, Canada, 2006, 1749-1752.

8. Ewerth, R., Mühling, M., Stadelmann, T., Qeli, E., Agel, B., Seiler, D., Freisleben, B. University of Marburg at TRECVID 2006: Shot Boundary Detection and Rushes Task Results. In *Online Proceedings of TRECVID Conference Series 2006*: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html

9. Gllavata, J., Ewerth, R., and Freisleben, B. Text Detection in Images Based on Unsupervised Classification of High-Frequency Wavelet Coefficients. In *Proceedings of 17th Int. Conference on Pattern Recognition*, Vol. 1, Cambridge, UK, 2004, 425-428.

10. Hu, Y. and Loizou, P. Subjective Comparison of Speech Enhancement Algorithms. In *Proc. Int. Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, Toulouse, France, 2006, 153-156.

11. Joachims, T. Transductive Inference for Text Classification using Support Vector Machines. In *Proc. of 16th International Conference on Machine Learning* (ICML), Bled, Slovenia, 1999, 200-209.

12. Kruizinga, P., and Petkov, N. Non-linear operator for oriented texture, *IEEE Trans. on Image Processing*, **8** (10), 1999, 1395-1407.

13. Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., and Duin, R. P. W. Limits on the Majority Vote Accuracy in Classifier Fusion. In *Pattern Analysis and Applications*, Vol. 6, No. 1, Springer-Verlag, London, 2003, 22-31.

14. Li, D., Sethi, I. MPEG Developing Classes. http://www.cs.wayne.edu/~dil/research/mdc/docs

15. Lienhart, R., Liang, L., and Kuranov, A. A Detector Tree of Boosted Classifiers for Real-time Object Detection and Tracking. In *Proceedings of IEEE Int'l Conf. on Multimedia & Expo, , Vol. 2*, Baltimore, Maryland, USA, 2003, 277-280.

16. Linguistic Data Consortium, The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus, 1990, available online (09.02.2007): http://www.ldc.upenn.edu/Catalog/readme_files/timit.readme.html.

17. Lu, L., Zhang, H.-J., and Li, S. Z. Content-based audio Classification and Segmentation by using Support Vector Machines. In *Multimedia Systems 8*, 2003, 482-492.

18. Lucas, B. and Kanade, T. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proc. of the Int'l Joint Conf. on Artificial Intelligence, Vancouver, Canada,* 1981, 674-679.

19. Snoek, Cees G.M., Worring, Marcel, van Gemert, Jan C., Geusebroek, Jan-Mark, and Smeulders, Arnold W.M. The Challenge Problem for Automated Detection of 101 Semantic Concepts in Multimedia. In *Proceedings of ACM Multimedia*, Santa Barbara, USA, October 2006, 421-430.

20. Truong, B.-T. and Venkatesh, S. New Enhancements to Cut, Fade and Dissolve Detection. In *Proc. of ACM International Conference on Multimedia*, Los Angeles, USA, 2000, 219-227.

21. Viola, P. and Jones, M. J.: Robust Real-Time Face Detection. In *International Journal of Computer Vision* 57(2), Kluwer Academic Publishers, Netherlands, 2004, 137-154.

22. Aslam, J.A., Pavlu V., and Yilmaz, E. Statistical Method for System Evaluation Using Incomplete Judgments, *Proceedings of the 29th ACM SIGIR Conference*, Seattle, 2006, 541-548.