

---

**Deliverable D1.2**

Visual, text and audio information analysis for hypervideo,  
first release

---

**Daniel Stein / Fraunhofer,**  
Jochen Schwenninger / Fraunhofer,  
Heike Horstmann / Fraunhofer,  
Evlampios Apostolidis / CERTH,  
Panagiotis Sidiropoulos / CERTH,  
Nikolaos Gkalelis / CERTH,  
Vasileios Mezaris / CERTH,  
Mathilde Sahuguet / Eurecom,  
Benoit Huet / Eurecom,  
Ivo Lašek / UEP,  
Tomáš Kliegr / UEP

28/03/2013

Work Package 1: Intelligent hypervideo analysis

**LinkedTV**

Television Linked To The Web

Integrated Project (IP)

FP7-ICT-2011-7. Information and Communication Technologies

Grant Agreement Number 287911

Dissemination level	PU
Contractual date of delivery	31/03/2013
Actual date of delivery	28/03/2013
Deliverable number	D1.2
Deliverable name	Visual, text and audio information analysis for hypervideo, first release
File	linkedtv-d1.2.tex
Nature	Report
Status & version	Final & V1.0
Number of pages	64
WP contributing to the deliverable	1
Task responsible	Fraunhofer
Other contributors	CERTH, Eurecom, UEP
Author(s)	<b>Daniel Stein / Fraunhofer,</b> Jochen Schwenninger / Fraunhofer, Heike Horstmann / Fraunhofer, Evlampios Apostolidis / CERTH, Panagiotis Sidiropoulos / CERTH, Nikolaos Gkalelis / CERTH, Vasileios Mezaris / CERTH, Mathilde Sahuguet / Eurecom, Benoit Huet / Eurecom, Ivo Lašek / UEP, Tomáš Kliegr / UEP
Reviewer	Jaap Blom / Beeld en Geluid
EC Project Officer	Thomas Küpper
Keywords	Multimodal Video Analysis, Shot Segmentation, Face Analysis, Video Concept Detection, Audio Analysis, Keyword Extraction, Named Entity Recognition, Video Event Detection, Object Re-detection

Abstract (for dissemination)	<p>Enriching videos by offering continuative and related information via, e.g., audiostreams, web pages, as well as other videos, is typically hampered by its demand for massive editorial work. While there exist several automatic and semi-automatic methods that analyze audio/video content, one needs to decide which method offers appropriate information for our intended use-case scenarios. We review the technology options for video analysis that we have access to, and describe which training material we opted for to feed our algorithms. For all methods, we offer extensive qualitative and quantitative results, and give an outlook on the next steps within the project.</p>
------------------------------	---

## 0 Content

<b>0 Content</b>	<b>4</b>
<b>1 Introduction</b>	<b>6</b>
<b>2 Shot Segmentation</b>	<b>8</b>
2.1 Problem statement and overview of the State of the Art	8
2.2 LinkedTV approach	8
2.3 Experimental evaluation and comparisons	10
2.4 Discussion	12
<b>3 Face Analysis</b>	<b>14</b>
3.1 Problem statement	14
3.2 Face detection	14
3.2.1 LinkedTV approach	14
3.2.2 Experiments: frontal face detectors	14
3.2.3 Experiments: profile face detectors	16
3.2.4 Optimization	17
3.2.5 LinkedTV choices for face detection	18
3.3 Spatio-temporal filtering	19
3.3.1 Problem statement	19
3.3.2 LinkedTV approach	19
3.4 Face clustering and recognition	19
3.4.1 Problem statement	19
3.4.2 LinkedTV approach	19
<b>4 Video Concept Detection</b>	<b>21</b>
4.1 Problem statement and overview of the State of the Art	21
4.2 LinkedTV approach	21
4.2.1 Video tomographs for concept detection	22
4.2.2 Base classifier fusion	23
4.3 Experimental evaluation and comparisons	25
4.4 Discussion	26
<b>5 Audio Analysis</b>	<b>27</b>
5.1 Speaker Identification	27
5.1.1 Problem statement and overview of the State of the Art	27
5.1.2 LinkedTV approach	27
5.1.3 Experiments	27
5.1.4 Discussion	27
5.2 Automatic Speech Recognition	27
5.2.1 Problem statement	27
5.2.2 LinkedTV approach	28
5.2.3 Simultaneous Perturbation Stochastic Approximation for ASR	29
5.2.3.1 SPSA Algorithm	29
5.2.3.2 Experiments for SPSA	30
5.2.3.3 WER optimization	30
5.2.3.4 Time-constrained WER optimization	32
5.2.4 Experimental evaluation of Automatic Speech Recognition	34
5.2.5 Discussion	34
5.3 Audio Fingerprinting	35
5.3.1 Problem statement and overview of the State of the Art	35
5.3.2 LinkedTV Approach	35
5.3.3 Experiments	36
5.3.3.1 Duplicate Detection	36
5.3.4 Discussion	38

<b>6</b>	<b>Keyword Extraction and Named Entity Detection</b>	<b>40</b>
6.1	Keyword recognition . . . . .	40
6.1.1	Initial version: all words . . . . .	40
6.1.2	Language-specific noun phrase chunking . . . . .	40
6.1.3	Statistical recognition with Stanford NER . . . . .	41
6.2	Keyword extraction . . . . .	42
6.3	Experimental results . . . . .	42
6.4	Extensions . . . . .	42
6.4.1	Identifying topical keywords . . . . .	43
6.4.2	Personalized keywords . . . . .	43
<b>7</b>	<b>Video Event Detection</b>	<b>45</b>
7.1	Problem statement and overview of the State of the Art . . . . .	45
7.2	LinkedTV approach . . . . .	45
7.2.1	Problem formulation . . . . .	45
7.2.2	Video representation . . . . .	46
7.2.2.1	Low-level visual features . . . . .	46
7.2.2.2	From low-level features to model vectors . . . . .	46
7.2.2.3	From frame-level to video-level representation . . . . .	46
7.2.3	Event detection . . . . .	46
7.2.3.1	Subclass divisions . . . . .	46
7.2.3.2	SRECOC framework . . . . .	47
7.3	Experimental evaluation and comparisons . . . . .	48
7.3.1	Dataset description . . . . .	48
7.3.2	Evaluation metric . . . . .	48
7.3.3	Experimental setup . . . . .	48
7.3.4	Results . . . . .	49
7.4	Discussion . . . . .	50
<b>8</b>	<b>Object Re-detection</b>	<b>51</b>
8.1	Problem statement and overview of the State of the Art . . . . .	51
8.2	LinkedTV approach . . . . .	52
8.3	Experimental evaluation and comparisons . . . . .	54
8.4	Discussion . . . . .	55
<b>9</b>	<b>Conclusion</b>	<b>56</b>
	<b>Bibliography</b>	<b>58</b>

# 1 Introduction

This deliverable presents the first release of visual, text and audio information analysis for hypervideo, as conducted and authored by WP1 of the LinkedTV project. Based on the state-of-the-art and requirement analysis in D1.1, the identified techniques will now be applied to the scenario content and their performance will be thoroughly evaluated.

A huge variety of techniques, both new and established, exist that analyze audio and video content (semi-)automatically. Ideally, the processed material then offers a rich and pervasive source of information to be used for automatic and semi-automatic interlinking purposes. However, the information produced by video analysis techniques is as heterogeneous as is their individual approach and their expected complexity, which is why careful planning is crucial, based on the demands of an actual use-case scenario.

In D1.1 we introduced the use-case scenarios in the LinkedTV project and gave a brief description of the main user-side requirements that arise from them. Two different types of scenarios were presented, the News Show and the Documentary scenario and for each scenario three different user archetypes were described. Through the analysis of the distinctive needs and demands of each user we pointed out the technical requirements from a user-side perspective, which helped us to define the different techniques that were utilized in order to provide the described services to the user.

The following sections dealing with the different techniques are structured in the same way: they start with a problem statement including a short review of the current state of the art. Afterwards we introduce the chosen approach for LinkedTV and present the results of experiments conducted for evaluation. Finally we discuss the evaluation results and thereby give an outlook on the next steps of our work.

We start in Section 2 with video shot segmentation which is used as a pre-processing step for multiple video analysis tasks. Shot segmentation techniques partition the video into elementary structural units, called shots, which are sequences of frames captured without interruption by a single camera. Essentially, shot segmentation can be seen as the basis of most high-level video content analysis approaches that are going to be developed in LinkedTV, being one of the major prerequisites for efficient semantic analysis, indexing and retrieval of video material. Section 3 concentrates on face analysis techniques which comprises three components: face detection, face clustering and face recognition. We present the three components in the order they are best applied to video material. Face detection comes first to give the temporal and spatial location of faces in the video. Afterwards face clustering is performed which enables to gather similar faces, i.e. faces that belong to the same person. Finally face recognition is presented, which enables matching a name with each face cluster.

We proceed in Section 4 with the technique of video concept detection which helps to automatically understand videos belonging to various domains. Especially the fast and accurate detection of concepts depicted in a video is still an essential and challenging problem. With respect to this, we present our approach and our current evaluation results. Subsequently we focus on audio analysis and its underlying technology in Section 5. The section starts with a general status on speaker identification and speech recognition systems, both optimized for German and Dutch. Then optimization approaches are presented to improve the performance of the German speech recognition system on spontaneous speech parts. Finally a novel method for audio fingerprinting is introduced. audio fingerprinting can be used for synchronisation of television content with second screen applications which is considered to be of interest for LinkedTV.

Section 6 deals with keyword extraction as a first step for the named entity recognition performed in WP2. There are several sources where we can retrieve textual information about a particular video, namely subtitles, annotations of videos (done by an author or an editor) or transcripts obtained as a result of automatic speech recognition. These texts are a valuable source of information about the video itself. Keyword extraction refers to the identification of important words within given textual information. These words are used to tag videos serving as descriptors for quick orientation in video content, easier filtering during searching and categorizing videos with the same tags.

Section 7 concentrates on video event detection as an advanced technique for more effective ways of indexing, summarizing, browsing, and retrieving of video content. In Section 8 an object re-detection approach is presented, which is designed for the identification of instances of manually selected objects in a video or a group of images, thus providing the medium for the automatic instance-based labeling of such content. Finally, we finish this deliverable with a short conclusion in Section 9.

## List of Figures

1	Examples of abrupt and gradual video shot transitions. . . . .	9
2	The overall scheme of the used shot segmentation algorithm. . . . .	10
3	An example of the file that is being processed by the flash detector. . . . .	11
4	The overall scheme of the extended shot segmentation algorithm. . . . .	12
5	Face detection results obtained with the CMU database, taken from [CDHL11] . . . . .	15
6	Profile face detection pipeline . . . . .	17
7	Face detection: frontal faces are in green bounding boxes, profile faces in pink ones . . . . .	18
8	The general pipeline of the employed concept detection system. . . . .	22
9	Two tomograph examples, each corresponding to a different type of tomograph image. . . . .	23
10	Performance comparison of a concept detection system. . . . .	26
11	Speaker identification for German politicians . . . . .	28
12	Example runs of SPSA and its word error rate progression on the development corpus. . . . .	31
13	WER and RTF results on the DiSCo corpora “clean planned” and “clean spontaneous”. . . . .	32
14	Optimization runs on the development set, with different RTF-penalized loss functions. . . . .	33
15	Scatter plot with all configurations, on the DiSCo test corpora. . . . .	33
16	Performance on the RBB Aktuell news show from 15th March 2011 . . . . .	33
17	Diagram depicting the audio fingerprint extraction algorithm. . . . .	36
18	Recall drop with increasing artificial noise level. . . . .	37
19	Recall drop with increasing passenger train noise level. . . . .	37
20	Content of four news shows containing material about Berlin’s new airport. . . . .	39
21	Current LinkedTV keyword recognition workflow. . . . .	41
22	The overall scheme of the initial implementation for object re-detection. . . . .	52
23	The overall scheme of the improved implementation for object re-detection. . . . .	53
24	Indicative examples of successful detection by the object re-detection algorithm. . . . .	55
25	The background noise that is imported by the current rectangular selection tool. . . . .	56
26	Performance comparison of the initial and the improved version of the object re-detection algorithm. . . . .	56

## List of Tables

1	Evaluation results of the shot segmentation algorithm. . . . .	11
2	Evaluation results of the baseline flash detector. . . . .	12
3	Frontal face classifiers performance on FDDB with 5171 faces . . . . .	16
4	Classifiers performance on 270 frames of SV seed video . . . . .	16
5	Profile face classifiers performance on FDDB with 5171 faces . . . . .	16
6	FAT classifier performance with variation in the number of minimum neighbors . . . . .	18
7	Summary of face classifiers performances on FDDB . . . . .	18
8	Free parameters of the decoding process. . . . .	30
9	WER and RTF results on all corpora, for the SPSA iterations and their respective loss functions. . . . .	31
10	ASR performance on RBB content. . . . .	34
11	Keyword extraction precision (p) and recall (r) on the various rbb segments, based on the ASR performance, for keywords with a focus on person names and places. . . . .	42
12	Evaluation performance on the TRECVID MED 2010 dataset using weak concept detectors. . . . .	49
13	Evaluation performance on the TRECVID MED 2010 dataset using strong concept detectors. . . . .	49
14	Evaluation performance on the TRECVID MED 2011 dataset using weak concept detectors. . . . .	50

## 2 Shot Segmentation

### 2.1 Problem statement and overview of the State of the Art

Video shot segmentation, also found in the literature as “shot boundary detection” and “shot transition detection”, is extensively used as a pre-processing step for multiple video analysis tasks, such as video classification, retrieval, video summarization and skimming, etc. In general, shot segmentation techniques aim to partition the video into consecutive frames captured without interruption by a single camera. These elementary structural units, which are called shots, by definition demonstrate a certain degree of temporal and visual affinity, thus constituting a self-contained visual entity. Based on this, it becomes clear that shot segmentation can be seen as the foundation of most high-level video content analysis approaches that are going to be developed in LinkedTV, validating it as one of the major prerequisites for efficient video semantic analysis, indexing and retrieval.

Since shots are defined as continuous temporal segments, shot segmentation can be handled as detection of the video shot boundaries, i.e. the temporal limits of each shot. The shots boundaries are determined by the type of transition that has been used at the stage of video editing. If the transition is abrupt (or “cut” as it is called in film grammar) the last frame of a shot is followed by the first frame of the next shot whereas if the transition is gradual (i.e. if an editing effect is used, like fade in/out, wipe, dissolve, etc.) there is a short intermediate temporal interval, in which the visual content of two consecutive shots is combined. An example of these two types of transition is presented in Figures 1(a) and 1(b).

Early shot segmentation approaches performed shot boundary detection based on pair-wise pixel comparisons between successive or distant frames of the video stream. In the last years a number of more sophisticated shot segmentation techniques can be found in the relevant literature, which can be roughly discriminated in two major categories: methods that use uncompressed video data and methods that are directly applied on the compressed video stream. A common approach of the first category employs color histograms and detection of shot boundaries based on the comparison of color histograms from successive frames, calculated either at the image level or at a more detailed block level [TTZ07]. Another alternative involves image structural features, like edges and performs shot detection by counting the Edge Change Ratio (ECR) [ZMM99] between successive frames of the video. Recent extensions of this idea combine edge information with color histograms [QLR<sup>+</sup>09] and motion [LL10]. In addition, a more elaborate approach that proposes the use of Support Vector Machine (SVM) classifiers has been introduced. This technique employs either pixel-wise comparisons, color histograms and edge information [LYHZ08], or image features that are not traditionally used for video shot segmentation like the Color Coherence [PZM96] and the Luminance Center of Gravity [TMK08]. Finally, following the introduction of powerful scale- and rotation-invariant local features like SIFT [Low04] and SURF [BETVG08], many authors presented several techniques that utilize these descriptors for video shot segmentation ([LZZ08],[BAB11]).

The techniques that belong to the second class, focusing mostly on reducing the computational complexity associated with processing at frame level, perform shot segmentation without including a prior video decompression into frames step. Such methods consider mostly MPEG video and exploit compression-specific cues to detect points in the 1D decision space where temporal redundancy, which is inherent in video and greatly exploited by compression schemes, is reduced. These cues can be macro-block type information of specific frames (e.g. intra-coded, skipped) [PC02] and DC coefficients or motion vectors that are included in the compressed data stream [DVZP04].

For a more detailed overview of the state-of-the-art techniques for shot segmentation, the reader is referred to Section 3.1 of D1.1.

### 2.2 LinkedTV approach

The employed technique used for the decomposition of the media content builds in the algorithm presented in [TMK08]. This algorithm takes as input an uncompressed video stream and performs both abrupt and gradual transition detection based on global and local visual information. More specifically, each frame is represented by a color histogram, based on the Macbeth color palette [MMD76], and a color coherence vector [PZM96], which is a two-dimensional color histogram vector that exploits both local and global colour information. Additionally, the technique employs the pixel’s intensity values, by estimating their spatial distribution expressed by the luminance center of gravity. Based on the above image features, the authors introduce three classification criteria for the detection of the shot boundaries





(a) An example of an abrupt transition between successive shots of a video stream. The camera stops capturing the last frame of the first shot and continues with the first frame of the following shot.



(b) An example of a gradual transition between successive shots of a video stream, in which the last frame of the first shot is gradually replaced by the first frame of the second shot. This type of video effect is called dissolve.

Figure 1: Examples of abrupt and gradual video shot transitions.

of the video, named respectively (a) Macbeth Color Histogram Change, (b) Color Coherence Vector Change and (c) Luminance Center of Gravity Change.

The overall process is demonstrated in Figure 2. After the extraction of the above features for each video frame (step 1), for every pair of consecutive or neighboring frames the distances between their feature vectors are estimated, thus forming the corresponding distance vectors (step 2). Subsequently, the distance vectors are joined into a unique 3-dimensional distance vector that is used as input to the meta-segmentation scheme, which is based on an SVM classifier. This scheme generates the overall response regarding the identification of a shot boundary between each pair of consecutive frames (step 3). The employed meta-segmentation scheme eliminates the need for threshold selection, contrary to what is typical in the relevant literature. Finally, it should be noted that due to the fact that the algorithm considers both pairs of successive and non-successive video frames, the algorithm can handle both cases of abrupt and gradual shot transitions.

While this approach was found in [TMK08] to outperform other state-of-the-art techniques, it seems that it suffers from over-sensitivity in cases where rapid changes in intensity occur within the same shot (e.g. caused by camera flashes). After examining the algorithm's performance in the video corpus provided by LinkedTV, we have found that many such instances are erroneously identified as shot boundaries. This flaw is emphasized in LinkedTV project, since one type of media content within the LinkedTV framework is news broadcasts, i.e. video streams where camera flash-lights appear in an increased frequency. Thus, it has become necessary to develop techniques that would overcome the problem of shot boundary mis-identification, due to flash-lights. For this purpose we extended the used shot segmentation algorithm with a baseline technique for flash detection.

The flash detector processes the intermediate results of the algorithm to refine the detected shot

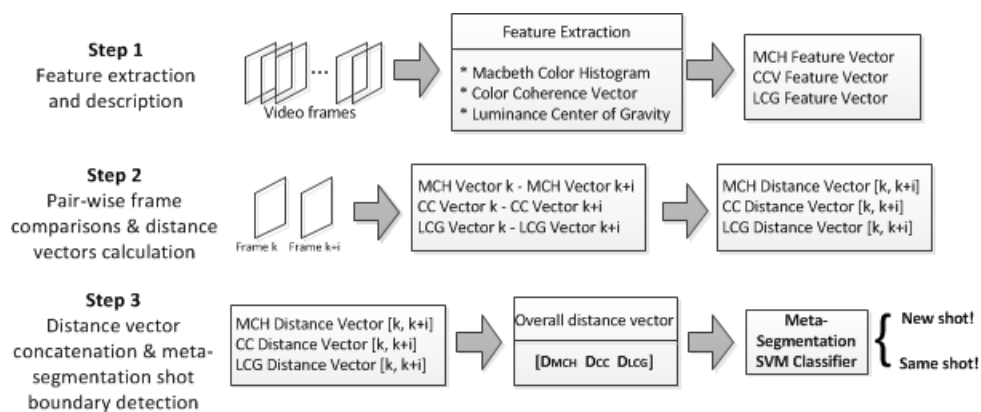


Figure 2: The overall scheme of the shot segmentation algorithm, described in [TMK08].

boundaries. More specifically, after the execution of the abrupt transition detection module a binary score, corresponding to the similarity/dissimilarity of consecutive frames, is assigned to each frame pair. As a matter of fact, a positive value (“1”) denotes that the two frames were found to have similar visual content, while a negative value (“-1”) that the visual content of the two consecutive frames differ significantly. Thus, a binary sequence is created, with length that is equal to the total number of video frames. Each value of this sequence corresponds to one frame of the video and the negative values indicate the positions (i.e. video frames) where abrupt color and/or intensity changes occur. An example of such sequence is demonstrated in the top of Figure 3 (part a), while the middle part (part b) shows a small part of such sequence where a shot boundary has been correctly detected; a new shot starts at the video frame that corresponds to the highlighted with green color value -1.

In order to avoid over-segmentation in cases of rapid camera movement, the algorithm arbitrarily selects that each video shot has a minimum duration of 25 frames (i.e. 1 second). However, by discarding for each shot boundary the next 24 frames the algorithm is unable to identify instances in which camera flashes are present. In these cases, usually a series of negative values are assigned to a short-term sequence of frames. Some indicative examples of such sequences are depicted in the bottom area of Figure 3, where the group of frames that have been affected by the camera’s flash-lights is presented with the red color.

Based on this analysis, we developed a baseline flash detector which post-processes the estimated set of shot boundaries to discard boundaries erroneously detected due to camera flashes. The novel algorithm use as input the sequence of binary values that has been calculated from the pair-wise comparison of consecutive frames of the video, and, by using pre-defined decision rules and manually selected temporal constraints, it detects the short-term groups of frames that correspond to camera flash-lights and discards the false shot boundaries. The overall shot segmentation algorithm, which includes flash detection, is demonstrated in Figure 4.

## 2.3 Experimental evaluation and comparisons

The evaluation of the shot segmentation algorithm’s performance was conducted using video content from the news show and the documentary scenario of the LinkedTV project. The ground-truth was manually generated, leading to 270 shots in the news show scenario and 446 shots in the documentary scenario.

The results summarized in Table 1, indicate that the algorithm performs remarkably good, since only few corrections need to be done manually after the automatic analysis of the video with the shot segmentation technique. More specifically, out of 270 actual shots from the videos of the news show scenario, the algorithm correctly detected 258, while 12 were missed (false negatives). Additionally, 11 shots were erroneously identified (false positives). On the other hand, from the 446 shots of the documentary scenario the algorithm correctly identified 416, while 30 were missed and 32 were erroneously detected. In both cases this small set of false positives and false negatives was caused due to rapid camera zooming operations (in or out) and shaky or fast camera movements.

Based on these results and trying to evaluate the algorithm’s performance, we calculated the precision and recall values. The first one measures the “quality” of the results by comparing the number



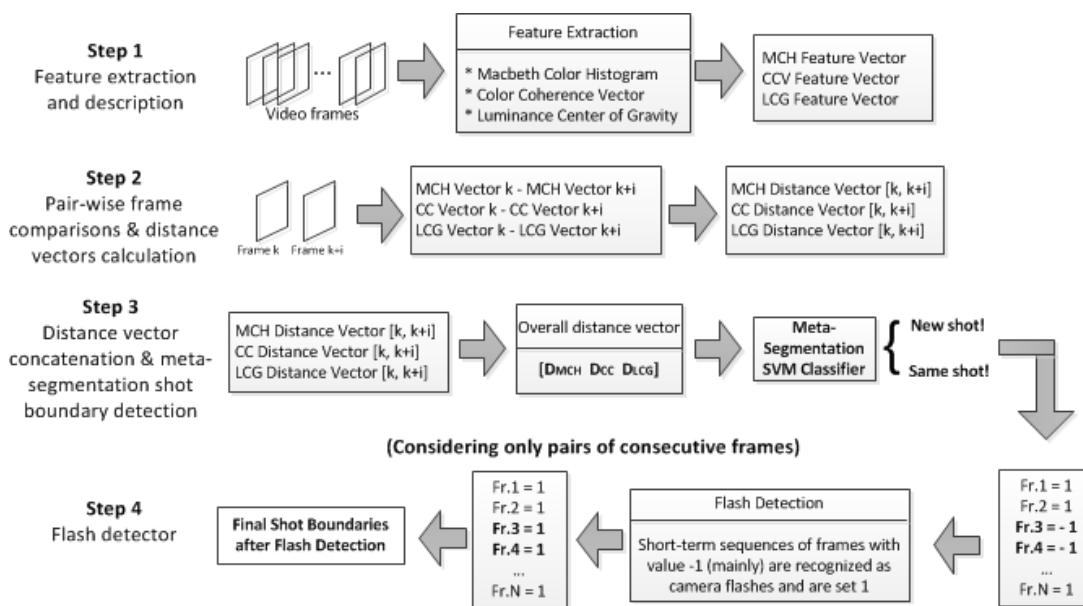


Figure 4: The overall scheme of the shots segmentation algorithm, which includes the baseline flash detector.

Table 2: Evaluation results of the baseline flash detector.

	Erroneous detections due to flash-lights
Without Flash Detection	18
With Flash Detection	4
Precision	1
Recall	0.78

recognized and eliminated. The precision of the flash detector is 1 since it didn't lead to any erroneous shot detection, while the recall is 0.78 which in other words means that the algorithm identified and corrected successfully around the 78% of the actual occurrences of camera flash-lights in the tested videos. These results indicate that the implemented baseline flash detector can contribute only positively to the improvement of the overall performance of the shot segmentation algorithm, by reducing the number of erroneous shot boundary detections due to camera flash-lights. It should be noted that we also tested flash detector to videos without camera flashes and no deterioration of performance was observed.

## 2.4 Discussion

Based on the analysis and the evaluation of the described technique for the temporal decomposition of videos into shots, as well as its extended version with the flash detector, we concluded that the algorithm already shows a remarkably good performance. However, it is clear that there is room for further improvement both in detection accuracy and time efficiency. Regarding the first goal, our future plans include further minimization of the algorithm's failures (both false positives and false negatives) and for doing so, we plan to implement an improved flash detector that will recognize and eliminate more efficiently the effect of camera flash-lights at the detection procedure. In addition we will try to find new ways to handle the erroneously detected shot boundaries due to fast movement and rapid zooming (in or out) operations of the camera. On the other hand, regarding time efficiency, the processing time for analyzing a video stream is found to be approximately 1,8 times of its actual duration (real-time). Our goal is to decrease the needed processing time in levels smaller than real-time. For this purpose, we intend to exploit the processing power of the modern Graphic Processing Units (GPU) in order to accelerate either the overall algorithm, or specific parts of it, by using e.g., a GPU-based algorithm for the

feature extraction-description step, or a GPU-based version of the SVM classifiers for the corresponding classification-detection step.

## 3 Face Analysis

### 3.1 Problem statement

Face analysis can be broken down into three components: face detection comes first, in order to give the temporal and spatial location of faces in the video; then face clustering enables to gather similar faces, i.e. faces that belongs to the same person. Last step is to perform face recognition in order to match a name with each face cluster.

### 3.2 Face detection

Face detection is the first stage of our analysis and it will impact to the outcome of other stages. Indeed, it is likely to present a non null error rate due to false positives and false negatives in the process. False positives will introduce noise in the clustering and recognizing steps, while false negatives are a miss. Hence, one of the main goals is to minimize this error and improve the robustness of the whole process. This is why serious attention has to be given to the detection task.

#### 3.2.1 LinkedTV approach

For face detection, we use the well-known Viola-Jones framework [VJ01], or more precisely its implementation in the C++ library openCV as improved by Lienhart and Maydt [LM02]. This method works for detecting faces in images, but we will describe in Section 3.3 how we adapt it for videos.

There are 3 phases in the Viola Johns technique, namely the following:

##### Feature Extraction

The Viola and Jones technique uses simple rectangular features, often called Haar-like features, that were extended by Lienhart and Maydt. Feature extraction is performed using what they called "integral images" for fast detection. We will also use a classifier based on LBP features, for recognition in a later stage.

##### Classification Using Boosting

Adaptive Boosting (AdaBoost) is a machine-learning algorithm that combines multiple simple classifiers (here, classifiers based on the cited Haar-like features) into a strong classifier by an iterative weighting scheme. At each stage, the classifiers are re-weighted accordingly to the error rate. This iterative procedure is stopped when all simple classifiers are trained. It is known to be less sensitive to overfitting than other learning algorithms.

##### Cascade and Multi-scale Detection

A cascaded classifier is used to combine many features efficiently. The classifier can be resized easily, so it can detect faces of different sizes in the image: this proves to be more effective than resizing the image. So, the classifier searches for faces in the image by gliding a fixed-size window every pixel of the frame to detect faces of specific size. After that, the window size is increased by a scaling factor and skimmed through the image again several times to detect all remaining faces in various sizes that may appear in the frame. A face is kept if there are at least  $n$  neighbors candidate rectangles.

Actually, openCV comes with several trained classifiers of this type, that have been trained using diverse training data, in different condition. Thus, a choice has to be done concerning the classifier to use. Moreover, Haar-like cascade classifiers can be parametrized with various settings: the initial window size, the scale factor and the number of minimum neighbors to retain a face. We will discuss them later in Section 3.2.4.

In the next sections we present the results of our tests that will help to choose the best classifier for our scenarios. Later in the document, we will perform further tests to tune the parameters of the classifier and to optimize the performances of the face detection.

#### 3.2.2 Experiments: frontal face detectors

As frontal faces and side-view faces have different appearances, different detectors have been trained for frontal and profile faces. We will first evaluate frontal face detectors only. For ease of use, we will take the following same notations as in [CDHL11] for the frontal face classifiers:

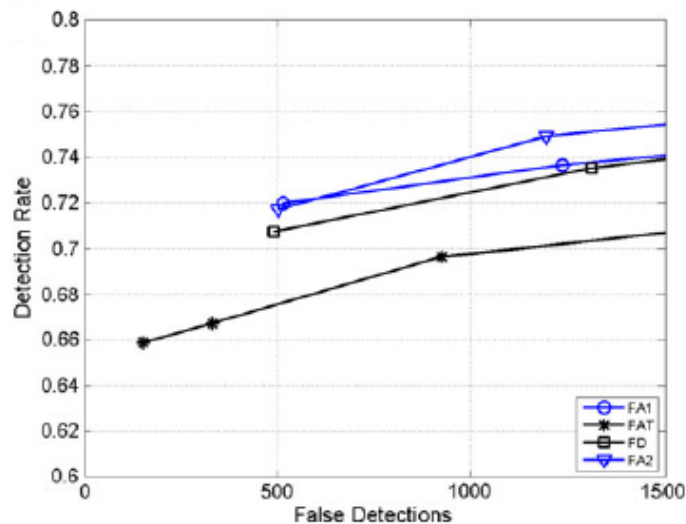


Figure 5: Face detection results obtained with the CMU database, taken from [CDHL11]

- haarcascade\_frontalface\_default : FD
- haarcascade\_frontalface\_alt : FA1
- haarcascade\_frontalface\_alt\_tree : FAT
- haarcascade\_frontal\_face\_alt2 : FA2
- lbpcascade\_frontalface : FLBP

In order to compare different classifiers and parameters, the most appropriate measure is the number of rightly and wrongly detected faces. Precision and recall can thus be calculated easily. As our goal is to recognize faces from videos, we would like to discard as much false positives as possible, in order to introduce as less noise as possible in the recognition step. Hence, we will aim to give priority to precision over recall.

In [CDHL11], the authors compare different classifiers on two image databases: the CMU dataset ([SK00]) and the Yale Face database ([BHK97]). While the Yale Face database contains images of faces in a constrained environment (frontal faces, same background, only the illuminations conditions differ), the CMU dataset presents images in conditions similar to what we find in our videos. Indeed, images were gathered from the Web and present faces appearing in different conditions, thus fitting better real-life situations.

For each frontal face classifier its receiver operating characteristic (ROC) curve was computed. The results are presented in Figure 5. The area under the curves seems to prove that FA1, FA2 and FD outperform FAT. FLBP was not evaluated.

In order to confirm those results and get detection rates instead of ROC curves, we assessed the classifiers on the Face Detection Data Set and Benchmark (FDDB) from [JLM10]. This dataset includes images from the Faces in the Wild dataset and contains 2845 images annotated with 5171 faces. This is much larger than the CMU dataset (721 faces). Next, this dataset will be used to choose the different parameters for our classifier by evaluating the different settings. The experiment was done using 3 neighbors, a scale factor of 1.1 and a minimum windows size of 40 pixels.

Table 3 presents the results in terms of true positive detection (TP), false positive (FP), false negative (FN) that were counted out of the detection results, and precision and recall that were calculated based on those counts. Obviously, true negatives don't exist for a face classifier. FA1, FA2, FAT and FLBP appear to have a high precision (superior to 0.9), which is of interest for us, with FAT having a precision of 0.980. We then have to balance those results with the recall rate: FA1 and FA2 both have a recall rate superior to 0.7, while FAT's recall is of 0.585. FLBP and FD are discarded because both their precision and recall are no better than FA1's. Unlike what was suggested in [CDHL11], FAT seems to be the most appropriate classifier to our needs.

Table 3: Frontal face classifiers performance on Fddb with 5171 faces

classifier	TP	FP	FN	precision	recall
FA1	3675	298	1496	0.925	0.711
FA2	3716	411	1455	0.900	0.719
FAT	3025	63	2146	0.980	0.585
FD	3670	1080	1501	0.773	0.710
FLBP	3398	365	1773	0.903	0.657
profiles	1580	2603	3591	0.378	0.306

Table 4: Classifiers performance on 270 frames of SV seed video

classifier	TP	FP	precision
FA1	203	38	0.842
FA2	225	33	0.872
FAT	202	9	0.957
FD	521	217	0.709

Last, we need to assess this choice on our seed videos: we tested those different classifiers on 270 frames retrieved from an episode of the Tussen Kunst & Kitsch show at the sampling rate of 1 frame per second. This evaluation was done manually, therefore only true positives and false positives were counted (no annotation was provided for the faces), and precision could not be calculated. Results are shown in Table 4. FAT has the highest precision by far as seen in Fddb dataset. FA1 and FA2 performs almost equally.

Observations on the behavior of these classifiers showed that FAT apparently detects only frontal faces while FA2 and FD can detect also side-view faces. Thus, FAT's recall could be improved if it was combined with a profile classifier with high precision (in order to keep a high precision for the combined classifier).

### 3.2.3 Experiments: profile face detectors

OpenCV contains a detector for right profile faces (haarcascade\_profileface.xml). It can be easily adapted to left profile faces by performing the detection on the vertically flipped frames. From now on, we will consider the profile detector as the combination of the right and left profile detections.

The detector performances on Fddb are displayed in Table 5. They are very low, so the classifier cannot be used by itself: it has to be combined with another classifier. The idea is to improve the robustness of the side-view face detection from both classifiers by reducing their false positives.

In a first stage, we choose to use a face classifier that could detect both frontal and profile faces as the second classifier to assess the presence of a face. As shown in Table 5, the results were interesting but this approach had several drawbacks: first, the processing time was a lot higher than for frontal faces only, because we applied another classifier to the entire frame. Also, this lead to ignore some profile faces because FD and FA2 were not designed for side-view faces, and thus re-detect a lot of frontal faces, while not being designed for this. Unfortunately, we cannot evaluate this behavior because we do

Table 5: Profile face classifiers performance on Fddb with 5171 faces

classifier	TP	FP	FN	precision	recall
profiles	2674	1509	2497	0.639	0.517
profiles + FD	2662	160	2509	0.943	0.515
profiles + FA2	2706	103	2465	0.963	0.523
profiles + eyes	2426	508	2745	0.827	0.469



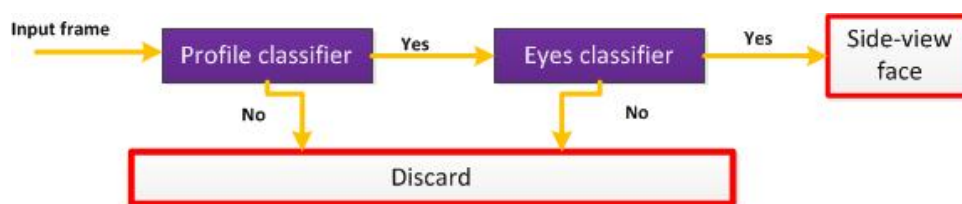


Figure 6: Profile face detection pipeline

not have an appropriate annotated dataset (with annotated profile faces).

Detecting an eye or an ear inside the detected face would greatly enhance the performance by increasing the probability that the given bounding box indeed contains a face. For a profile face, at least an eye and an ear should be detected. Thus, we performed tests using the profile detector and then further filter the results by running eyes detection on the returned faces. This method uses several classifiers and thus increases the processing cost as said earlier. Nevertheless, as we only perform the search for facial elements inside a small percentage of the initial frame, this effect is greatly reduced.

The pipeline of profile face detection is illustrated in Figure 6.

### 3.2.4 Optimization

#### De-interlacing

Before analysis, videos have to be preprocessed to be suitable for efficiency concerns. In this detection phase, video de-interlacing technique and video size adjustment are performed when necessary to improve the robustness of the algorithm.

First, interlacing is a technique that digital cameras and digital broadcasting use to double the perceived frame rate without consuming extra bandwidth. However, interlaced recordings result in side-effects in frame processing. Therefore, the first step in this video-based face detection phase is to de-interlace the video.

#### Video Size Adjustment

Furthermore, an appropriate video size (size of the frames) has to be chosen, in relation with the size of Haar-like feature window. If the video size is too large then it is intensively computational costly. Conversely, if the video size is too small, it is less efficient for the Haar-like cascade classifier to process. In this case, the rate of false negatives may increase since the Haar-like feature window is relatively large with respect to the frame size.

#### Minimum face size criteria

In order to reduce the false positive rate in face detection, too small faces are removed. In a video sequence, if a face having substantially small size compared with the frame size then it is likely a false positive and should be removed. Thus, the minimum size of face detection is set to a certain percentage of the frame size (currently 7%). The detection window won't have a size smaller than that, saving some computations. This is setting the initial windows size.

#### Scale factor

This factor determines by which factor the search window is scaled between the subsequent scans. We kept the default parameter of 1.1 which is a balance between the processing time (the bigger the scale factor, the less iterations, the faster the detector) and error rate (a search window increasing quickly may miss faces of intermediate dimension). 1.1 means increasing window by 10%.

#### Number of minimum neighbors optimization

The number of minimum neighbors `minNeighbors` is a parameter specifying how many neighbors (detected faces rectangles) each candidate rectangle should have to retain it. A small value of `minNeighbors` will cause a lot of faces to be detected for the same person (bounding boxes differing by a few pixels) while a too big `minNeighbors` will have the effect of missing some faces. With such a parameter higher than 0, isolated boxes, that are more likely to be false positives, are discarded. Table 6 compares the different results of face detection with the chosen frontal classifier FAT depending on the

Table 6: FAT classifier performance with variation in the number of minimum neighbors

minNeighbors	TP	FP	FN	precision	recall
0	3519	64405	1652	0.052	0.680
1	3284	213	1887	0.939	0.635
2	3143	103	2028	0.968	0.608
3	3025	63	2146	0.980	0.585
4	2907	38	2264	0.987	0.562
5	2800	28	2371	0.990	0.541

Table 7: Summary of face classifiers performances on Fddb

classifier	TP	FP	FN	precision	recall
combination used for LinkedTV	3462	162	1709	0.955	0.670
FAT	3025	63	2146	0.980	0.585
profiles	2674	1509	2497	0.639	0.517

value of minNeighbors. 4 neighbors seems to be a reasonable choice because it balances a high precision and an average recall.

#### Efficiency criteria

The computational cost for detection has to be taken into account. [CDHL11] evaluates the processing time on the CMU dataset for the classifiers: they are all comparable, ranging from 60.4 to 70.8. As it is similar for the cited classifiers, this criteria did not have any influence on our choice.

#### 3.2.5 LinkedTV choices for face detection

After running the aforementioned tests, we could design a face detector that was a combination of frontal and profile face classifiers.

We used the frontal face classifier with a minimum face size of 7% of the frame size, 4 neighbors to keep a face and a scale factor of 1.1.

The profile classifier was the combination of profile classifiers (left and right) and eyes classifiers. If eyes are detected within the face box, the face is retained. The same parameters as the frontal face classifier were used, except for the minimum size of detection for the eyes that was set to zero and we set the minimum neighbors parameter to 3.

The results are given in Table 7 and compared to simple FAT and profile classifiers. Image 7 illustrates the behavior of the classifier: frontal faces are boxed in green rectangle while profile faces are boxed in pink rectangles.

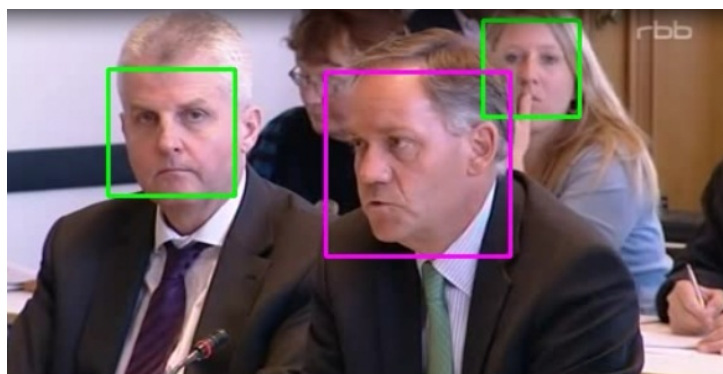


Figure 7: Face detection: frontal faces are in green bounding boxes, profile faces in pink ones

A drawback of using multiple classifiers is the processing cost. While a single classifier can run more or less real-time (depending on the classifier and the settings used), running several classifiers multiplies the processing time. A distributed implementation can be used to reduce it: each classifier (frontal, right profile and left profile) runs on a different machine; at the end a function is used to combine all results. Hence, parallelizing tasks would enable the detection to run at the speed of the slowest classifier (the profile ones, because they need a filtering step with another classifier). This will be studied in future work.

**Output of the detection** The output of the process is a xml file that contains face information at the frame level. Frames are grouped into shots. A face is referred to by the location of its bounding box: coordinates of the top-left pixel (x,y), width w and height h of the box.

### 3.3 Spatio-temporal filtering

#### 3.3.1 Problem statement

The tested framework performs well on images, we now need to adapt it to videos. Indeed, we can use spatio-temporal information present in a video shot in order to smooth the results.

After detection is made on every frame of the video, we make a second pass through the result xml file. We aim at making face tracks (linking faces of a person within a shot) that will benefit both detection and clustering processes: first, building face tracks enable to add missed faces by interpolating results; second, building face tracks is a first phase of the clustering process.

#### 3.3.2 LinkedTV approach

For this purpose, we will follow the work of [KMSZ10]. Face detections are linked between frames using a KLT feature tracker [ST94] using agglomerative clustering. Face tracks may have some missing faces, which can be interpolated thanks to temporal support between frames: if frames number n and n+2 contain a faces at almost the same position, it is more likely that frame n+1 also contains a face that was not detected and should be added. On the contrary, if a frame is the only one of a shot to contain a face at a given location, it is more likely that the face is a false positive that should be removed.

The output of this process is a set of face tracks that is a great input for clustering because it may already handle a variability in faces appearance (different poses, illumination conditions, etc for a same person).

### 3.4 Face clustering and recognition

#### 3.4.1 Problem statement

While face detection is pretty mature, face clustering and recognition techniques are still work in progress and need to be further studied and experimented. It is the next focus of our work. First, we will group faces of the same person appearing in a video into cluster. This person will be given an identifier, so to be able to retrieve all faces from this person. Later, the recognition module will enable to give a name to this person, and thus match the identifier to a real-world name: some extra information is needed, like an external annotator or knowledge of already labeled images (that can be stored in a database and/or mined from the web).

#### 3.4.2 LinkedTV approach

##### Face pre-processing

In order to analyze only the face features, we will crop the face images in order to remove background pixels. Facial features will be used when possible.

##### Clustering process

Face tracks (Section 3.3) will be available as a first input for face clustering. Next step is to group faces from different shots together. As said in the previous deliverable, we plan to use Local Binary Patterns (LBP) features to describe faces. PCA analysis will enable to reduce dimensions of the data. Then, we will study both an iterative clustering algorithm and K-means clustering algorithm. We will keep in mind that we seek precision over recall, the goal being to have clusters as pure as possible.

**Recognition**

As recognition is putting a label on faces, future work comes down to creating a database of models of people likely to be recognized. Following audio analysis approach in Section 5.2.2, we intend to gather from the web or the end-user partners images corresponding to persons likely to appear in the shows. This work will start by registering all persons who appear frequently (anchors, reporters, experts, etc), and important personalities (e.g. German politicians). Then, this database will grow along with the processing, when a new person will be annotated with his/her name. Also, we will make use of people information in the shows when available (metadata, casting, etc) in order to select a subset of persons to look for in that particular show.

## 4 Video Concept Detection

### 4.1 Problem statement and overview of the State of the Art

One of the main goals of the image and video processing community is to develop techniques that would allow the automatic understanding of unconstrained video. By exploiting this kind of information, groups of videos as well as links between them can be established, thus contributing to the envisioned interactive and interlinked television. In the root of this task lies the fast and accurate detection of the concepts depicted in the video. The efficient and effective detection of concepts by looking purely at the visual content is an important and challenging problem.

In the last years, the research community, partially triggered by the TRECVID Semantic Indexing task [SOK09], has shifted its focus on large-scale video concept detection, i.e. the development of systems that would be able to handle large amounts of video data and detect multiple semantic concepts efficiently (e.g. [WPZ12], [SSL<sup>+</sup>11]). As a result, several powerful techniques have emerged, aiming to compromise between high precision and low computational cost. For example, in order to exploit color information in addition to local image structure, the Opponent-SIFT and RGB-SIFT (or Color-SIFT) variations of the well-known SIFT descriptor [Low04] were proposed in [SGS10]. Furthermore, in order to reduce computational cost, SURF [BETVG08] and DAISY [TLF08] descriptors were introduced as fast SIFT approximations; interest point detection (traditionally performed with the help of corner detectors, e.g. the Harris-Laplace one [HS88]) was fully or partially replaced in many schemes by dense sampling (i.e. the sampling of image patches on a regular dense grid); and chi-square kernels, that were originally considered to be optimal for use in SVMs [ZMLS07a], [JNY07] are now often replaced by Histogram Intersection kernels [MBM08] or even Linear SVMs, to name a few recent developments in this area.

Contrary to what is intuitively expected, in most of the developed schemes that aim to detect multiple concepts in video data, motion information is ignored and the detection is based exclusively on a set of characteristic key-frames that are extracted at shot level (i.e. each video shot is represented by one or more key-frames). This is explained by the fact that motion descriptor extraction is typically associated with high computational cost, and the gains in precision that are attained by introducing motion descriptors in the concept detection process are often disproportionately low, compared to the added computational complexity. However, a concept detection algorithm that uses no motion information handles the video stream as a mere collection of photos (key-frames), failing to take advantage of the dynamic nature of video that makes it particularly expressive.

Similarly, most techniques that involve more than one classifier for each concept perform a fusion by linear combination of the probability values (e.g. in [NRT<sup>+</sup>11]) or even by averaging (e.g. [SSL<sup>+</sup>11]). On the contrary, all the sophisticated fusion schemes introduced so far have failed to improve the classification accuracy, compared to a plain averaging of the classifier results. For example, both the linear regression and the dimensionality reduction that were proposed in [HRBO11] and [DPG<sup>+</sup>11] respectively were found to have almost equal accuracy with corresponding approaches that used averaging.

### 4.2 LinkedTV approach

We have tried to overcome the motion descriptors high-computational cost by using spatio-temporal slices with one axis in time and one in space, called video tomographs [TA94]. These tomographs are straightforwardly extracted, their extraction requires extremely low computation cost, and, as it is demonstrated, they can then be analyzed as if they were plain images. We report that video tomographs, when used along with visual key-frames, enhance video concept detection while being a computationally efficient solution towards exploiting information about the temporal evolution of the video signal.

Moreover, we have started to explore approaches that would successfully replace the averaging of all classifiers with a more sophisticated fusion scheme and introduced a generic methodology that builds upon the results of a genetic algorithm, thus controlling which sub-set of the available classifiers should be combined for developing an optimal detector for each specific concept. Preliminary experimental results manifest that the proposed approach both enhances the accuracy and reduce the overall computational cost.

The pipeline of the employed concept detection system is shown in Figure 8. The video stream is initially sampled, generating for instance one or multiple key-frames per shot. Subsequently, each sample is represented using one or more types of appropriate features (e.g. SIFT [Low04], SURF [BETVG08], etc.). These features form the input to a number of base classifiers, which use vector quantization and SVMs. The parameter sets that control the employed classifiers are predefined (i.e.

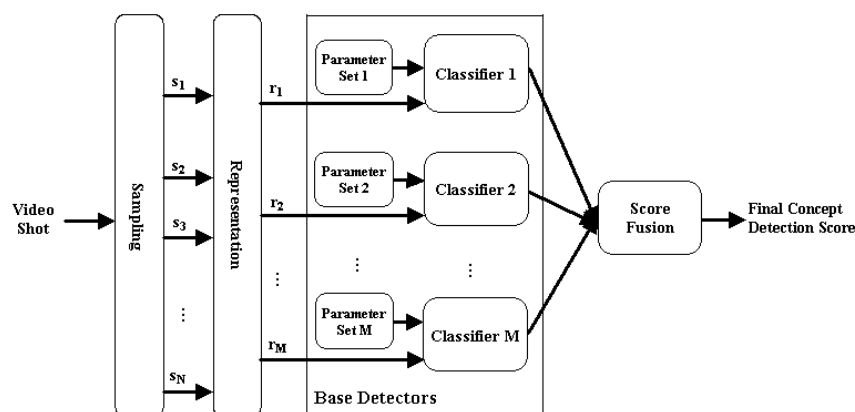


Figure 8: The general pipeline of the employed concept detection system. Initially the video stream is sampled (e.g. key-frames are extracted) using  $N$  different sampling strategies (labeled  $s_1, s_2, \dots, s_N$  in the figure). Subsequently,  $M$  sets of features are extracted to represent the visual information samples (labeled  $r_1, r_2, \dots, r_M$  in the figure). The set of features are used as inputs to base classifiers that are trained off-line. Finally, the base classifier outputs are combined and an overall concept detection score is estimated.

they have been learned at the classifier training stage), using similar features extracted from training data. Finally, the base classifier outputs are fused to estimate a final concept detection score. It should be noted that this process is executed multiple times, independently for each one of the considered concepts that are to be detected.

The most interesting parts of this methodology relate to the first and the last component of the analysis pipeline, i.e. the video sampling, to extract not only key-frames but also video tomographs, and the sophisticated combination of the base classifier outputs. Apart from these novelties, all other components have been built following well-known state-of-the-art approaches. More specifically, we have employed SIFT, RGB-SIFT and Opponent-SIFT image descriptors in our system, which were experimentally found (see [SGS10]) to form the optimal low-level visual descriptor set for video concept detections tasks. These descriptors are extracted from local image patches. Similarly to the current state-of-the-art, two approaches for selecting these patches are used. In the former the interest points are selected through dense sampling, while in the latter interest point detection is performed through a Harris-Laplace corner detector [HS88]. The extracted low-level descriptors are assigned to visual words using separately two vocabularies that were created off-line through k-means clustering, employing hard-assignment and soft-assignment respectively [GVSG10]. A pyramidal  $3 \times 1$  decomposition scheme, employing 3 equally-sized horizontal bands of the image [LSP06], is used in all cases, thus generating 3 different Bag-of-Words (BoWs) from image bands, while a fourth BoW is built using the entire image. In all cases, the number of words for each BoW was set to 1000. Thus, for each combination of video sampling strategy, interest point detector, descriptor and assignment method a vector of 4000 dimensions is finally extracted and used as the actual input to the utilized base classifiers. The latter are linear SVMs, chosen so as to significantly reduce the required computation time. All classifiers were trained off-line, using the extensive training data that is provided as part of the TRECVID 2012 Semantic Indexing task [OAM<sup>+</sup>12].

#### 4.2.1 Video tomographs for concept detection

In this subsection we discuss about the improvement of keyframe-based concept detection by augmenting the set of key-frames with a spatio-temporal type of image, the video tomograph. Video tomographs were introduced in [TA94] as spatio-temporal slices and have been used for optical flow estimation [HS95], camera motion classification [JL12] and video copy detection [LKF09], [MKNR12]. A video tomograph is defined in [TA94] as a cross-section image, i.e. an image defined by the intersection between a plane and the video volume. The cross-section image is generated by fixing a 1-D line on the image plane and aggregating the video content falling on the corresponding line for all frames of the shot.

The two most simple tomograph images are the centralized horizontal (CH-tomograph) and the centralized vertical (CV-tomograph) tomographs. A CH-tomograph is constructed by aggregating for all frames of a shot the visual content of the horizontal line passing from the frame center. A CV-tomograph

is constructed in an analogous way, with the only difference being that the line is perpendicular to x-axis, instead of parallel. In Figure 9 a CH-tomograph and a CV-tomograph example are shown. In the left example the shot shows the national anthem ceremony in a sports event. As the camera follows the raising flag, the CH-tomograph “draws” a flipped version of the scene background. The flipping artifact is not expected to play an important role in the following steps of the concept detection algorithm, since most of the well-known low-level descriptors are orientation invariant. On the other hand, in the right example the video shot depicts a city square. In this case, the camera is moving in the horizontal direction. The CV-tomograph, which is generated by lines perpendicular to the camera motion direction, generates a “mosaic-like” image of the urban scene.



Figure 9: Two tomograph examples, each one corresponding to a different type of tomograph image. The left tomograph is a CH-tomograph, while the right a CV-tomograph. Both of them are defined by the temporal ordering of lines that pass from the center of the frame. Three indicative frames of the shot from which each tomograph was generated are also shown to the left of the corresponding tomograph (the temporal order of the shown frames is from the top to the bottom).

For the purpose of concept detection, the tomographs are processed in the same way as key-frames. More specifically, image patches are estimated, followed by descriptor extraction and vector quantization. It should be noted that the vocabulary employed at this stage is constructed by clustering visual words extracted from the corresponding tomograph type (e.g. a random sample of CV-tomograph SIFT vectors is clustered in order to generate the vocabulary used for vector quantization of descriptors extracted from CV-tomograph images). The resulting BoW feature vectors are the input to tomograph-based base classifiers. These classifiers are also independently trained for each tomograph type, using annotated samples taken from tomographs of the corresponding type. Finally, the base classifier output is fused with the output of the keyframe-based classifiers in a simple averaging scheme that does not discriminate between outputs of key-frame and tomograph-based classifiers.

#### 4.2.2 Base classifier fusion

The pipeline of Figure 8 involves multiple configurations that are executed independently, prior to combining the intermediate results. Such a design is justified by the fact that a system that aims to detect a large number of concepts should be able to handle concepts that demonstrate significant diversity. For example, the concept set that we are using to evaluate our system includes 346 concepts. Among them there are concepts that are either static (e.g. “forest”) or dynamic (e.g. “running”), specific (e.g. “George Bush”) or generic (e.g. “building”), human-based (e.g. “two people”), object-based (e.g. “motorcycle”) or background-based (e.g. “static background”), characterized by the audio content (e.g. “singing”) by the visual content (e.g. “nighttime”) or both (e.g. “explosion”), etc. Thus, a multiple-concept detection scheme is expected to include a number of base classifiers, each one contributing to the accuracy enhancement of a certain class of concepts. The approach used to fuse the base classifier outputs is examined in this subsection.

As it is already mentioned, the most common strategy is to merge probability estimations, using either averaging or linear combination with weights that are globally tuned for all the employed concepts. The latter approach suffers from the “curse of dimensionality” that prohibits a brute-force tuning, especially since typically a large amount of classifiers is used (e.g. 25 classifiers in the LinkedTV approach). Moreover, both averaging variations do not take into account the fact that many base classifiers focus only on certain concept classes. Consequently, they can be discarded from all other concepts, thus reducing the overall computational complexity. As a matter of fact, assuming that the classification is

performed in shot level, the associated computational complexity of the detection of concepts in a video is  $O(S * D * C_{gl})$ , where  $S$  is the total number of video shots,  $D$  is the amount of concepts and  $C_{gl}$  is the (constant) number of classifiers used for each concept.

Instead of this, we propose a scheme that determines independently the sub-set of base classifiers that will be employed for each concept. Subsequently, for each one of them the results of all classifiers that belong to the corresponding sub-set are averaged. The selection scheme is a two-step algorithm, with the first step being a genetic algorithm and the second step being a novel post-processing approach. It should be noted that this process takes place during training and is executed off-line. As a result, for each concept the classifiers that are selected not to participate in the optimal sub-set are excluded from the corresponding detection scheme, thus reducing the associated complexity to  $O(S * \sum_{i=1}^D C_i)$ , where  $C_i$  is the number of classifiers employed to the detection of the concept with index  $i$ .

The genetic algorithm that is executed in the beginning of this approach is summarized in Algorithm 1.

---

**Algorithm 1** Concept detection post-processing genetic algorithm.

---

Notation:  $c$  is the current concept,  $L$  the ordered set of classifiers,  $L_i$  a subset of this set,  $\#$  the operator used for set cardinality,  $m$  the mutation rate of the genetic algorithm,  $N$  the number of initial random samples,  $R$  the number of repetitions,  $k$  the number of non-discarded subsets in each step,  $p_i$  the performance achieved by using the average of the configurations that belong to  $L_i$  and  $v_i$  the participation vector of subset  $L_i$ . As participation vector of a subset  $L_i$  we refer to a binary vector of length  $\#L$ , which has 1 in the  $j$ -th dimension if and only if the  $j$ -th element of  $L$  belongs to  $L_i$ .

- 1: Initially, from set  $L$ ,  $N$  random subsets  $L_1, L_2, \dots, L_N$  are selected and the corresponding participation vectors  $v_1, v_2, \dots, v_N$ , as well as the corresponding performance estimations  $p_1, p_2, \dots, p_N$  are computed. The current iteration index  $r$  is set to 1.
  - 2: The  $k$  random subsets that achieved the best performance “survive”, while all the other subsets are discarded.
  - 3: The  $k$  “survived” random subsets are combined in  $k * (k - 1) / 2$  pairs to breed two random subsets each, thus leading to a new subset pool of  $k(k - 1)$  members. From two random subsets  $L'_i$  and  $L''_i$  the children subsets will have participation vectors  $v'_i * v''_i + Y * (1 - v'_i * v''_i)$  and  $v'_i * v''_i + (1 - Y) * (1 - v'_i * v''_i)$ , where all operations are boolean and  $Y$  is a random binary vector of dimension  $\#L$  and with  $(\#L) / 2$  1s.
  - 4: From the  $k(k - 1)(\#L)$  dimensions of the generated subset pool,  $mk(k - 1)(\#L)$  change value (from 0 to 1 or from 1 to 0).
  - 5: The subsets that match to the resulting  $k(k - 1)$  generated participation vectors are retrieved and the corresponding performance is estimated.
  - 6: If  $r = R$  then the subset  $L_j$  that achieved the maximum performance is returned as the optimal configuration selection. Moreover, the participation vectors  $v_1, v_2, \dots, v_T$  of the subset that achieved the top- $T$  accuracy are retrieved. Otherwise,  $r = r + 1$  and the algorithm continues from step 2.
- 

At this stage the algorithm may be terminated, retrieving the subset that achieved the maximum performance as the one selected for the specific concept. However, experimental evidence in multiple datasets demonstrated that the subset that corresponds to the maximum performance in a training dataset does not always lead to performance improvement in a validation dataset. As a matter of fact, as it will be shown in the next subsection, in terms of mean concept detection accuracy, an approach that employs such a post-processing scheme is at the same levels as a technique that does not employ it, but instead estimates the average classifier score of all classifiers. Therefore, we have developed a novel algorithm that further processes the genetic algorithm outcome, before retrieving the actual sub-set of classifiers that will be used to determine the detection scheme output.

This algorithm is based on experimental evidence that signify the relation between the probability that the sub-set suggested by the genetic algorithm will increase the accuracy and the size of the sub-set that the genetic algorithm suggests. Apparently, the larger this sub-set is the higher the probability that this sub-set will lead to an accuracy increase if used for the detection of the specific concept. Consequently, we have developed a top-bottom approach, which starts from the selected sub-set being the complete set of classifiers and iteratively generating pruned versions of it. At each iteration larger sub-set “parts” are allowed to be discarded, but on the other hand at each iteration the discard employs more strict criteria. Finally, the sub-set “parts” that are allowed to be discarded are determined by common classifier specifications (for example, all classifiers that use Harris-Laplace detector and soft assignment or all classifiers that use RGB-SIFT descriptor, CV-tomographs and hard assignment). This algorithm is summarized in Algorithm 2.



**Algorithm 2** Concept detection post-processing pruning algorithm.

Notation:  $C$  is the set of configurations,  $T_0$  is the initial threshold,  $r$  the threshold increase ratio,  $l$  is the iteration index,  $T_l$  the threshold at iteration  $l$ ,  $v_1, v_2, \dots, v_T$  the participation vectors of the subset that achieved the top- $T$  accuracy,  $d$  the length of each participation vector,  $P_i$  is the  $i$ -th partition of  $C$  into non-overlapping subsets,  $P_{ij}$  the  $j$ -th sub-set of the  $i$ -th partition,  $v_{ij}$  the participation vector of  $P_{ij}$ ,  $S_{ij} = \sum_{k=1}^{k=T} (\sum v_{ij} * v_k) / (T * \sum v_{ij})$ ,  $A_l$  is the set of active sub-sets at iteration  $l$ ,  $Q_l$  is the set of query sub-sets at iteration  $l$ .

- 0: Initialization:  $T_1 = T_0$ ,  $A_1 = \emptyset$ ,  $Q_1 = \{C\}$
- 1: For all query sub-sets  $q_i$ , for each  $P_j$ , the intersection  $q_{ij} = q_i \cap P_j$  and the score  $S_{ij}$  are estimated.
- 2: If  $\max(S_{ij}) < (T_l * S_i)$  for all  $P_j$ , then  $q_i$  is moved from the query set to the set of active sub-sets, else all  $q_{ij}$  except the one with the lower  $S_{ij}$  are added to the query set.
- 3: If  $Q_l = \emptyset$  the algorithm pass to step 4, else  $l = l + 1$ ,  $T_l = r * T_{l-1}$  and the algorithm continues from step 1.
- 4: The retrieved sub-set is the one that has participation vector the union of the participation vectors of all active sub-sets.

### 4.3 Experimental evaluation and comparisons

To examine the contribution of tomographs towards more accurate concept detection, we conducted an experimental comparison of a concept detection scheme that employs only 1 key-frame per shot and a concept detection scheme that additionally employs 1 CH-tomograph and 1 CV-tomograph per shot. We selected these two simple tomographs for our experiments in order to demonstrate that tomographs can enhance performance even if a non-optimized, simple tomograph extraction method is followed. Additionally, a third configuration in which only the aforementioned tomographs are used was also included in the comparison. The experimental setup employs the entire video dataset and the concept list that were used in the 2012 TRECVID SIN task.

More specifically, 46 semantic concepts were evaluated. The detection of these concepts takes place in a video dataset comprising 8263 videos of almost 200 hours total duration. The whole dataset is off-line pre-segmented into more than 140 thousand shots. The goal of each concept detector is to retrieve the top-2000 shots that are most likely for the concept to be present. The 2000 shots are sorted using the detectors' score in descending order and the results are evaluated using partial, manually generated ground-truth annotations. The employed detection accuracy measure is Extended Inferred Average Precision (xinfAP) [YKA08], which is a measure approximating Average Precision, when the ground-truth annotations are not complete. The employed ground-truth annotations and the xinfAP implementation are the ones provided by the TRECVID organizers.

The experimental results are shown for each concept in Figure 10. Although many of the 46 concepts are not intuitively expected to be strongly correlated with any type of motion (e.g. "landscape", "fields", "computers") we can see from this figure that combining key-frame- and tomograph-based concept detection increases the accuracy for 39 of the 46 concepts. Overall, the performance as measured by mean xinfAP increases from 0.135 to 0.156, representing a 15.5% accuracy boost. This, together with the standalone performance of video tomographs, which is expressed by a mean xinfAP of 0.044, show that although the tomographs are not potential replacements of the key-frames, they provide additional information that the latter do not capture, thus being a valuable addition to key-frame-based concept detection approaches.

Furthermore, these results indicate that using tomographs in addition to one or a few key-frames is beneficial, compared to using a large number of key-frames for each shot. In [SSL<sup>+</sup>11], a concept detection scheme similar to our baseline keyframe-based approach was employed, in two versions differing only in that the first one exploited only 1 key-frame for each shot, while the second employed 10 additional key-frames. The accuracy boost achieved by the second version in relation to the first one was 14.7%, which is comparable to the one achieved by our approach through the introduction of a pair of tomographs, but the associated computational cost of using an extra 10 key-frames per shot is higher than the cost of using a pair of tomographs by one order of magnitude.

Finally, it should be noted that the concepts that benefit the most from the introduction of tomographs are, as expected, the dynamic concepts, i.e. those that are clearly related with motion. In the employed concept dataset we have identified 15 concepts that are either directly related with actions that involve motion (e.g. "throwing", "walking-running", "bicycling") or objects that are very likely to be filmed while they are in motion (e.g. "skier", "motorcycle", "boat-ship"). In Figure 10 these concepts are marked with a "\*\*". If only these concepts are taken into account, the accuracy boost caused by introducing

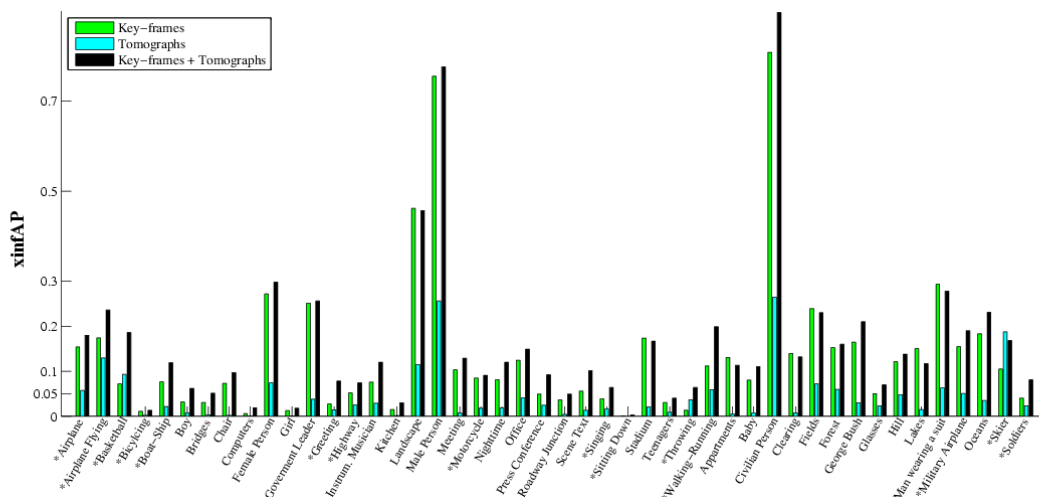


Figure 10: Performance comparison of a concept detection system that uses tomographs plus key-frames versus a system that uses exclusively key-frames, and a technique that uses exclusively tomographs, in TRECVID 2012 Semantic Indexing dataset. Concept detection accuracy is measured by xinfAP.

tomographs is 56.4% (mean xinfAP rising from 0.074 to 0.116). For the remaining, rather static concepts, the corresponding mean xinfAP boost is limited to 11%.

Regarding the fusion technique presented in Subsection 4.2.2, we can not report anything more than preliminary experimental results, since the parameter tuning of Algorithm 2 is still under development. Currently, when using the same experimental setup with the one used to evaluate tomograph contribution the mean accuracy increases from 0.156 to 0.164, while the computational complexity gain is 29%. It should be noted that if only the genetic Algorithm is used the mean accuracy is almost equal (actually, it slightly drops from 0.156 to 0.154), while the computational complexity gain is 66%.

#### 4.4 Discussion

This period our work focused on the use of video tomographs as an additional sampling strategy for the video concept detection task, as well as the development of a novel technique to fuse the base classifier results. While the fine-tuning and further testing of the base classifier fusion technique that we propose in this section is still work in progress, it seems that by adopting such a scheme that combines video tomographs with intelligent selection of base classifiers at the individual concept level it is possible to simultaneously reduce significantly the computational cost and increase the concept detection accuracy.

## 5 Audio Analysis

### 5.1 Speaker Identification

#### 5.1.1 Problem statement and overview of the State of the Art

Speaker identification (SID) aims at recognizing persons based on their voice. Towards that end, SID systems usually employ a two-step approach. In the first step, called the enrollment, a new speaker is added to internal database and a statistical model representing the characteristic voice features is constructed. Once this step is done, this person can be distinguished from other speakers automatically by scoring utterances against all available models, and normally a special “unknown speaker” model.

The models for the speakers are often based on Gaussian Mixture Models (GMMs), with features capturing the spectral properties of a voice via Mel-Frequency Cepstral Coefficients (MFCCs), and sometimes high-level speech information such as pronunciation variations, prosody, idiolect or characteristic conversational topics [RQD00],[RAC+03],[PNA+03],[AKC+02],[D+01].

In recent years, modeling the speakers via Maximum-Likelihood Linear Regression (MLLR) features [MB02] or Eigenvoices [TKNJ00] has become popular, especially if only sparse training data is available. Additional efforts are required when trying to separate the influences of channel, background, and speaker [Ken05], [KL10].

#### 5.1.2 LinkedTV approach

For speaker identification, we follow the well-known approach of [RQD00], i.e., we make use of Gaussian Mixture Models (GMMs) using spectral energies over mel-filters, cepstral coefficients and delta cepstral of range 2. An overall universal background model (UBM) is merged from gender-dependent UBMs and forms the basis for the adaptation of person-dependent SID models.

#### 5.1.3 Experiments

We listed German politicians as a possible set of persons to be identified based on the scenario description. Thus, we downloaded a collection of speeches from 253 German politicians, taken from the archive of the German parliament.<sup>1</sup> In total, this consists of 2581 files with 324 hours of training material. To make training of the models feasible, we use 2 minutes per file to adapt the UBM.

In the seed videos from the news show analyzed, no German parliament speaker was present. Since we are looking for reliable results on a large data set, we took a distinct set of 994 audio files taken from German parliament speeches to evaluate the quality of the models. Speaker Recognition evaluation is given as the equal error rate (EER), i.e., the error for the rejection threshold which produces an equilibrium of false positive and false negative matches. We also depict the Detection Error Trade-Off (DET) curves as described in [MDK+97]. A GMM with 128 mixtures has an Equal Error Rate (EER) of 9.86, whereas using 1024 mixtures leads to an improvement of 8.06 EER. See Figure 11 for Detection Error Trade-Off (DET) curves.

#### 5.1.4 Discussion

As demonstrated with the experimental results, the performance of the speaker identification component given a reasonable training corpus is very promising. Our main problem is the collection of speaker labeled material from the local context of both the news show and the documentary scenario. Face (re-)detection is only a weak indication of a person speaking, since for example in interview situations the camera will often focus on the face of the conversational partner to show his or her emotion to the viewer. Also, from our experiences NER is of little help here because a person seldom says his own name in interview situations.

## 5.2 Automatic Speech Recognition

### 5.2.1 Problem statement

Automatic Speech Recognition (ASR) describes the process of automatically converting spoken words into text. Typical large-vocabulary systems capable of recognizing conversational speech (as opposed

<sup>1</sup><http://webarchiv.bundestag.de/cgi/archive.php>, accessed: 28/03/2013

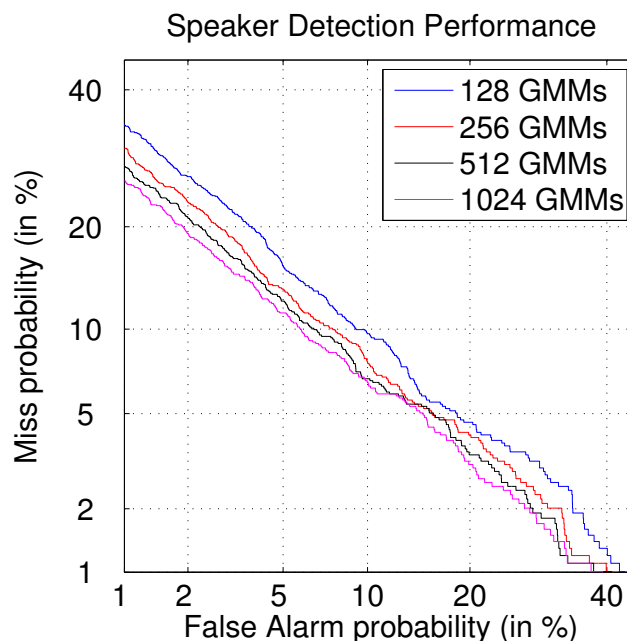


Figure 11: Speaker identification for German politicians: DET Curve for different mixture sizes of the GMM, on a withheld test corpus of 994 audio files from the German parliament.

to command and control applications with only relatively few possible commands) are built upon three main information sources:

**Acoustic model** The acoustic model contains the statistical representation of the features extracted from the audio stream and phonemes or triphones, which are essentially the building blocks of speech.

**Dictionary** The dictionary defines the set of words that can be recognized and contains the pronunciation alternatives, thus mapping phonemes to actual words.

**Language model** The language model assigns probabilities to sequences of words (n-grams), therefore modeling the typical use and phrases in a language.

### 5.2.2 LinkedTV approach

For German speech recognition, we employ a state-of-the-art speech recognition system as described in [SSE08]. For training of the acoustic model, we employ 82,799 sentences from transcribed video files. In accordance with the news show scenario, they are taken from the domain of both broadcast news and political talk shows. The audio is sampled at 16 kHz and can be considered to be of clean quality. Parts of the talk shows are omitted when, e.g., many speakers talk simultaneously or when music is played in the background. The language model consists of the transcriptions of these audio files, plus additional in-domain data taken from online newspapers and RSS feeds. In total, the material consists of 11,670,856 sentences and 187,042,225 running words. Of these, the individual subtopics were used to train trigrams with modified Kneser-Ney discounting, and then interpolated and optimized for perplexity on a with-held 1% proportion of the corpus.

For Dutch, the SHOUT speech recognition toolkit as described in [OHdJ<sup>+</sup>09] is used that deploys speech activity detection (speech/non-speech) and speaker segmentation/clustering preceding multiple decoding passes involving feature normalization using among others vocal tract length normalization (VTLN), speaker cluster adapted acoustic modeling, and optionally language model adaptation or lattice-rescoring. The models employed in SHOUT have not been adopted in any special way except for a monthly updated language model that reflects changes in every-day use of vocabulary, which is most probably not crucial for this material.

Since we expect a Berlin dialect, we further crawled the web for lexica, which are often offered from tourist sites. Merely introducing new words would render them unknown to the language model, which

is an unwanted effect. While the grammar of the dialect can of course vary at great length, the sentence structure often has very similar patterns (notable exceptions include the permutation of dative and accusative pronouns in the Berlin dialect). Thus, we try to introduce the new words as pronunciation variants of High German words whenever possible. For some more complex words, e.g., “Erbbejrbnis”, literally “heritage funeral” which refers to restaurants with very frequent changes of ownership, we introduce it internally as a pronunciation variant for a word that can be used in the same semantical context (in this case: “restaurant”). In total, the new vocabulary consists of 501 words most frequently used in the Berlin dialect.

### 5.2.3 Simultaneous Perturbation Stochastic Approximation for ASR

The current free parameters in the speech recognition system have been optimized for news speech and the system thus performs poor in the spontaneous parts. To overcome this, we tried to optimize them on a corpus that contains an equal mix of spontaneous and planned speech.

Both the optimization of the acoustic model and the language model in automatic speech recognition for large vocabularies are well-established tasks. Reducing the perplexity of the language model on a withheld development set, for example, is a common way to achieve lower word error rates (cf. [KP02]). The actual decoding process, however, also uses a large set of free parameters that have to be adopted to the given task or domain. While some parameters directly weight the models, others affect the size of the search space, where it is even harder to estimate the effect on the hypothesis quality and on the expected decoding time.

In praxis, these parameters are often set empirically in a rather tedious task, which is even more complex whenever a real-time factor (RTF) constraint has to be fulfilled. Moreover, they should be adopted to new domains, whenever the training material changes, or when more sophisticated decoding servers are available that could possibly allow for either faster decoding or better decoding in the same amount of time.

In LinkedTV, we employ Simultaneous Perturbation Stochastic Approximation (SPSA) [Spa92] for the overall optimization of the free decoding parameters and will show that it leads to stable and fast results. Further, we show that by extending the loss function that has to be optimized with a RTF penalty, arbitrary time constraints can be fulfilled while maintaining a best-possible output quality automatically. This is especially interesting for large databases or applications that have to run on hardware-restricted architectures. We offer our results on selected RBB material as well as on the German Difficult Speech Corpus (DiSCo) [BSB<sup>+</sup>10] corpus.

To the best of our knowledge, no automatic optimization technique for the free parameters during the decoding phase in automatic speech recognition is explained in the literature. The decoders typically offer rough ranges and default values for their parameters (e.g., in HTK [YEG<sup>+</sup>06], Julius [LKS01] or Kaldi [PGB<sup>+</sup>11]). The Sphinx [WLK<sup>+</sup>04] Wiki<sup>2</sup> offers quite detailed ways on how to improve the performance speed, but again the methods have to be manually adopted to the task.

In the field of machine translation (MT), the free parameters of recent decoders (e.g., [KHB<sup>+</sup>07, VSHN12]) are typically estimated either with the Downhill Simplex Method [NM65] or with Och’s Minimum Error Rate Training [Och03]. SPSA has been employed for MT as well and has been shown to be much faster in convergence than downhill simplex, while maintaining comparable hypothesis quality results [LB06].

SPSA has already been applied to various tasks other than natural language processing, such as statistical simulations, traffic control, as well as signal and image processing [Spa98b].

**5.2.3.1 SPSA Algorithm** For the optimization of a tuple of free parameters  $\theta$ , we employ the SPSA algorithm [Spa92], which works as follows:

Let  $\hat{\theta}_k$  denote the estimate for  $\theta$  in the  $k$ -th iteration. Then, for a gain sequence denoted as  $a_k$ , and an estimate of the gradient at a certain position denoted as  $\hat{g}_k(\cdot)$ , the algorithm has the form

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \hat{g}_k(\hat{\theta}_k) \quad (3)$$

In order to estimate  $\hat{g}_k(\cdot)$ , we perturbate each  $\hat{\theta}_k$  with a vector of mutually independent, mean-zero random variables  $\Delta_k$ , multiplied by a positive scalar  $c_k$ , to obtain two new parameter tuples:

<sup>2</sup><http://cmusphinx.sourceforge.net/wiki/sphinx4:largevocabularyperformanceoptimization>, accessed: 30.1.2013

Table 8: Free parameters of the decoding process. Some parameters are given individually to the 1<sup>st</sup> pass or 2<sup>nd</sup> pass of the Julius decoder, and are marked with (2). Continuous parameters are marked by a trailing .0

name	start	min	max
(2) LM weight	10.0	0.0	20.0
(2) ins. penalty	-7.0/10.0	-20.0	20.0
(2) beam width	250/1 500	700/20	3000/1000
score envelope	80.0	50.0	150.0
stack size	10 000	500	20 000
#expanded hyp.	20 000	2 000	20 000
#sentence hyp.	10	5	1 000

$$\hat{\theta}_k^+ = \hat{\theta}_k + c_k \Delta_k \quad (4)$$

$$\hat{\theta}_k^- = \hat{\theta}_k - c_k \Delta_k \quad (5)$$

For a loss function  $L(\cdot)$ , we then estimate  $\hat{g}(\hat{\theta}_k)$  as:

$$\hat{g}(\hat{\theta}_k) = \begin{bmatrix} \frac{L(\hat{\theta}_k^+) - L(\hat{\theta}_k^-)}{2c_k \Delta_{k1}} \\ \vdots \\ \frac{L(\hat{\theta}_k^+) - L(\hat{\theta}_k^-)}{2c_k \Delta_{kp}} \end{bmatrix} \quad (6)$$

We follow the implementation suggestions in [Spa98a] and use a  $\pm 1$  Bernoulli distribution for  $\Delta_k$ , and further set:

$$a_k = \frac{a}{(A+k+1)^\alpha} \quad \text{with } a = 2, A = 8, \alpha = 0.602$$

$$c_k = \frac{c}{(k+1)^\gamma} \quad \text{with } c = 0.25, \gamma = 0.101$$

**5.2.3.2 Experiments for SPSA** For optimization, we chose to optimize both parameters that primarily affect the search space as well as those that affect the internal weighting/penalty of the underlying models. On the one hand, some settings might require more internal hypotheses to fully take effect, on the other hand, the search space directly affects the RTF which we also want to optimize.

For developing, we use a corpus from German broadcast shows, which contains a mix of planned (i.e., read news) and spontaneous (i.e., talk shows) speech, for a total of 2348 utterances (33744 words).

For evaluation, we make use of clean speech segments of the DiSCO corpus as described in [BSB<sup>+</sup>10], and use “planned clean speech” (0:55h, 1364 utterances) as well as “spontaneous clean speech” (1:55h, 2861 utterances).

Table 8 lists the Julius parameters, the ranges that we allow for as well as the starting values for optimization. Internally, we map these ranges to  $[-15 \dots +15]$  for the SPSA iterations. If the parameters are integers, we store them as floats internally but truncate them for each loss function call.

**5.2.3.3 WER optimization** First, we optimized the parameters on the word error rate (WER), i.e., the number of substitutions, insertions and deletion errors divided by the reference length. Preliminary experiments showed that a percentage value resulted in a gradient too low for a meaningful update in Eqn. 3. We thus multiplied the WER by a factor of 100 so that it should range between 0 and 100 instead of 0 and 1.

The results on the development set are shown in Figure 12. In total, the hypothesis quality improved by 1.9 WER absolute (6.4 rel.). In a second run (s. Table 9), the improvement was similar and converged after 10 iteration runs already. The results on the test sets are presented in Figure 13. It can be seen that the optimization generalizes nicely on both DiSCO corpora: 1.2% WER absolute improvements on the

Table 9: WER and RTF results on all corpora, for the SPSA iterations and their respective loss functions. Each optimization for the unconstrained and delta loss function has been executed two times from scratch to check for convergence.

loss function	iteration	dev		test planned		test spontaneous	
		WER	RTF @1.6GHz	WER	RTF @2.6GHz	WER	RTF @2.6GHz
baseline	0	29.6	5.3	24.0	4.6	31.1	4.0
unconstrained	18	27.7	?	22.8	5.4	28.4	5.9
unconstrained	18	27.7	7.3	22.6	6.1	28.4	6.1
delta	18	27.6	5.3	22.2	4.5	27.7	4.8
delta	18	27.6	?	22.5	4.2	27.9	4.4
increasing	14	32.5	3.0	26.1	2.2	31.9	2.3
	+28	31.6	2.9	25.3	2.5	30.0	2.6

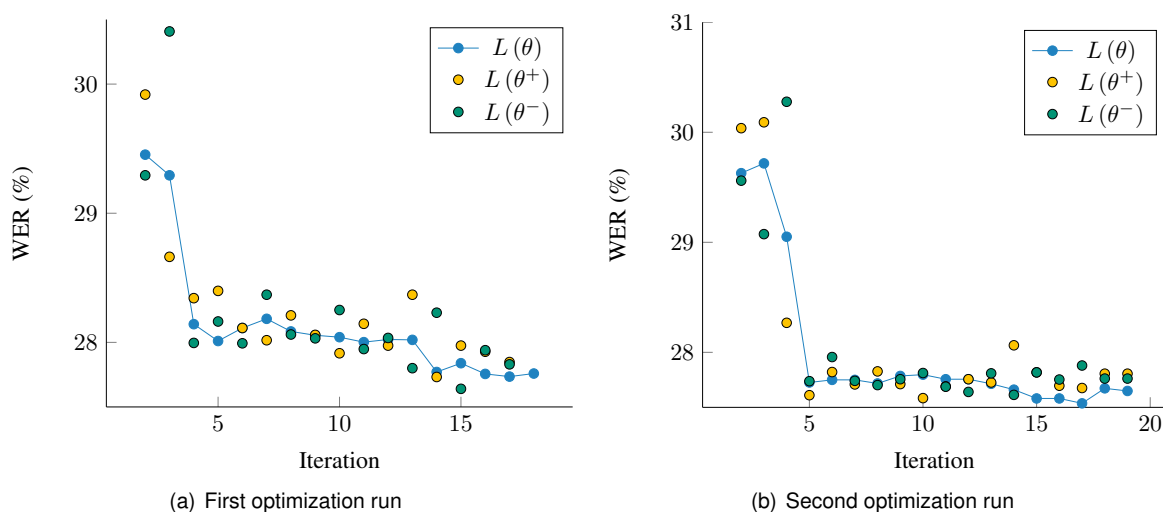
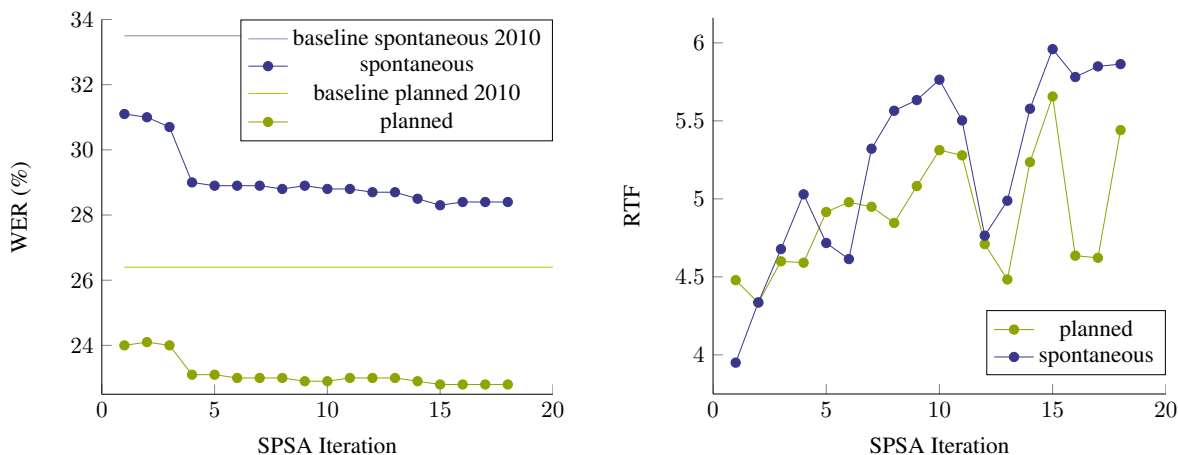


Figure 12: Example runs of SPSA and its word error rate progression on the development corpus.



(a) WER progression on the baseline given has been extracted from [BSB+10]. (b) RTF development on the DiSCO corpora “clean planned” and “clean spontaneous”, for the first optimization run.

Figure 13: WER and RTF results on the DiSCO corpora “clean planned” and “clean spontaneous”.

planned speech task, and 2.7% WER absolute improvement on the spontaneous speech task (s. Figure 13(a)), over a strong baseline surpassing the results given in the original corpus paper [BSB+10]. However, these improvements come with a rather high price in terms of RTF (s. Figure 13(b)). While for many settings this might not pose a problem, in time-crucial applications this is not desirable. Thus, in a second set of experiments, we try to take the RTF into account.

**5.2.3.4 Time-constrained WER optimization** In these sets of experiments, we penalize the loss function by a RTF dependent term  $\mu$ :

$$L(\hat{\theta}_k) = \text{WER}(\hat{\theta}_k) + \mu(\hat{\theta}_k). \tag{7}$$

It soon became apparent that careful planning is needed in order to obtain the desired result. Intuitively, we penalized RTFs exponentially, which turned out to deteriorate the parameters too much when the initial RTF was already substantially above this given threshold. This was especially a problem for optimization on a slow machine, where the WER completely deranged (i.e., drop of 30% absolute) due to a severe gradient misjudgement in the first iteration.

Instead adding the delta of the actual RTF to the WER,

$$\mu(\hat{\theta}_k) = \text{RTF}(\hat{\theta}_k), \tag{8}$$

lead to an equilibrium (s. Figure 14(a)), a trend that was reproducible on a second optimization run (s. Table 9). In general, the RTF appeared more stabilized, with no loss in WER visible.

In a final experiment, we penalized the RTF increasingly with each iteration:

$$\mu(\hat{\theta}_k) = (\text{RTF}(\hat{\theta}_k)) \cdot \tilde{k}, \tag{9}$$

with an increasing  $\tilde{k} = k$  as long as a RTF threshold is not reached. For the first iteration where the RTF factor is equal the threshold,  $\tilde{k}$  is fixed in order to give the optimization the ability to converge, thus stabilizing the WER. In our experiments, we arbitrarily set the RTF threshold to 3, which was reached in iteration 12 and 14, respectively. After this, the WER stabilized. Figure 14(b) depicts one iteration run, Table 9 shows all results.

In order to see whether our optimization is a reasonable trade-of between RTF and WER, we collected all results from the iterations and computed their convex hull (Figure 15(a)). It can be seen that the final SPSA iteration for each optimization run (marked by filled-out symbols) is typically part of the convex hull or very near to its border. From our optimization runs, we could see no gain for the RTF-unconstrained loss function. A delta RTF penalized loss function could result in a configuration that performs better in terms of WER and is generally faster. If the RTF is penalized increasingly in each step, the WER rate is still within reasonable range for a much more comfortable RTF.

The results on the RBB content are shown in Figure 16. In general, the findings of the DiSCO corpus carry over nicely. With the delta approach, the WER has improved 2.8% absolute (7.2% relative) while maintaining a comparable RTF.



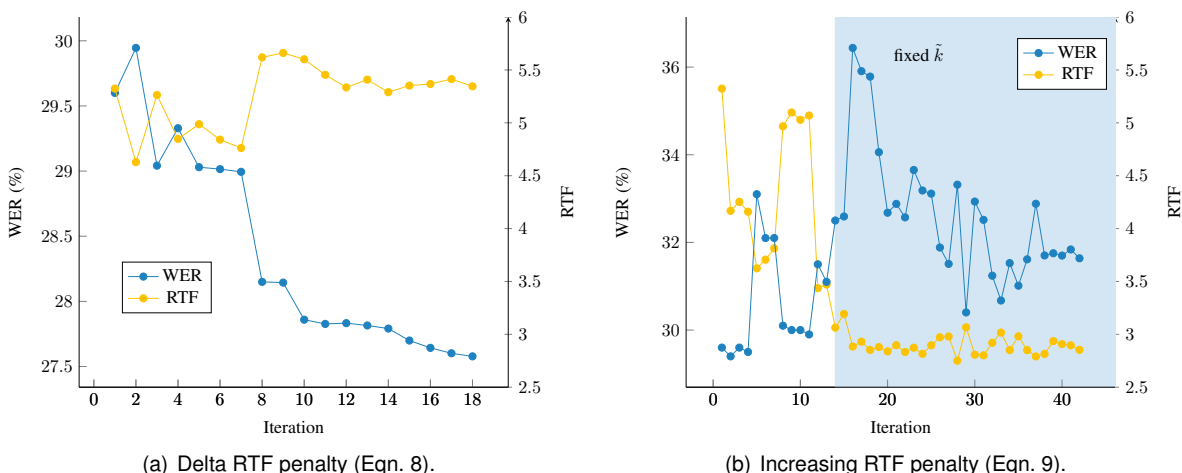


Figure 14: Optimization runs on the development set, with different RTF-penalized loss functions.

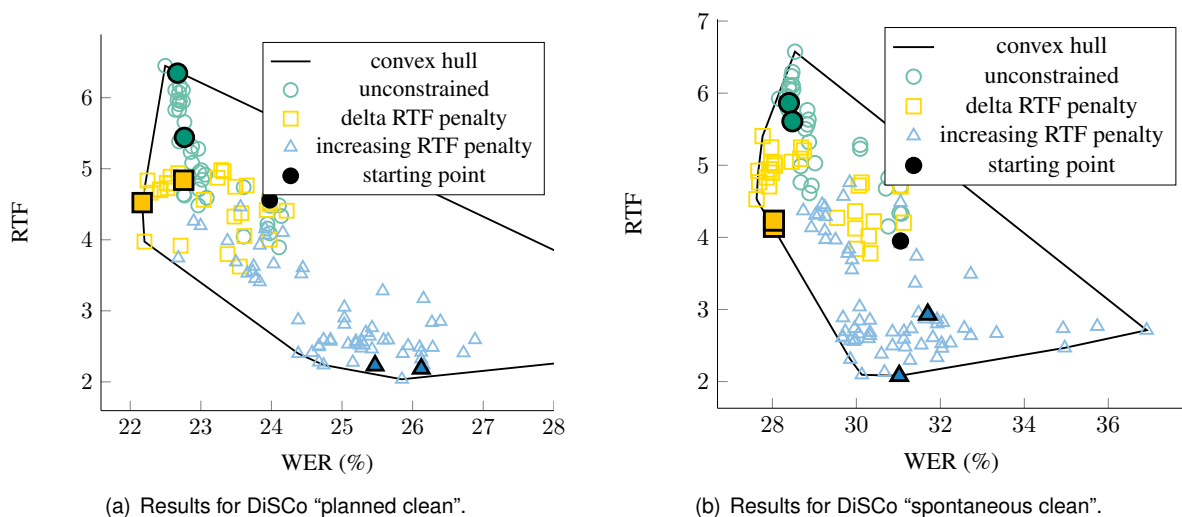


Figure 15: Scatter plot with all configurations, on the DiSCO test corpora. The final optimization iteration is marked by filled-out symbols.

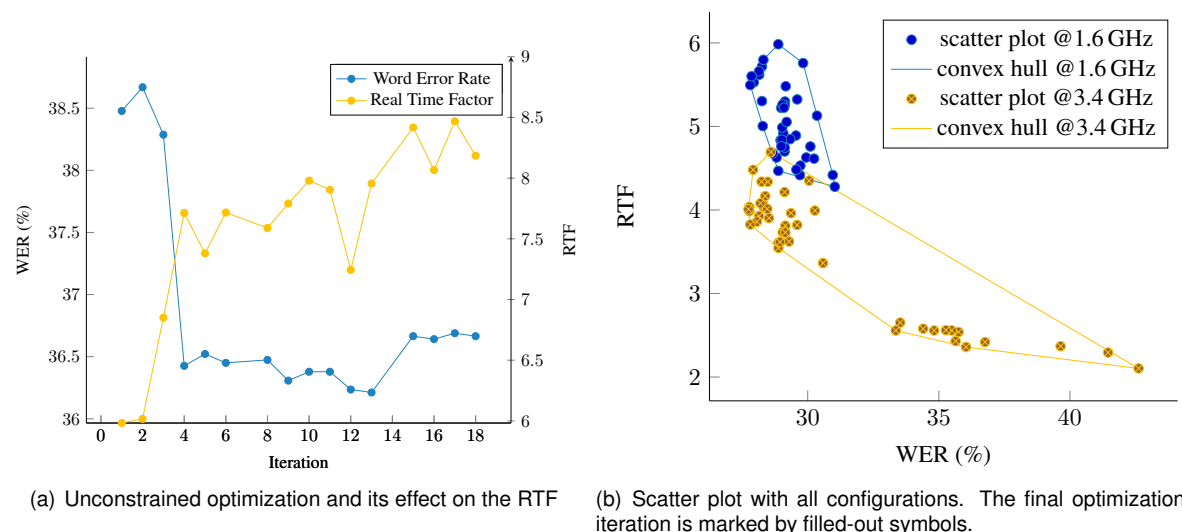


Figure 16: Performance on the RBB Aktuell news show from 15th March 2011

Table 10: ASR performance on RBB content.

segment	ASR performance (WER)
new airport	36.2
soccer riot	44.2
various other news I	9.5
murder case	24.0
boxing	50.6
various other news II	20.9
rbb challenge	39.1
weather report	46.7

#### 5.2.4 Experimental evaluation of Automatic Speech Recognition

Speech recognition is commonly measured as the word error rate (WER), which is defined by the Levenshtein distance [Lev66] (i.e., the minimum number of substitutions, deletions, and insertions necessary to transform the hypothesis into the reference), divided by the reference length. On the news material, we annotated one video of half an hour length. The German ASR system had an overall WER of 38.5%, with the largest error source being substitutions (25.6%). See Table 10 for a more fine-granular WER analysis, based on the segment. While some parts where a trained moderator is speaking have an error rate of as low as 9.5%, other segments featuring many spontaneous interviews with background noise from the street (like boxing and soccer riot) are much worse. Please note that these result do not include optimized parameters via SPSA.

Introducing the local pronunciation variants as described above gave 1% absolute improvement for the relevant parts. However, a proportion of locals speaking with a dialect with heavy background noise (Berlin tavern visitors talking about a local soccer team, 9 utterances in total) is absolutely not intelligible. From the scenario point of view, the last case is unsatisfactory. We assume that all three archetypes of the news show scenario are locals, and that we will loose substantial information if the dialect prohibits ASR access. However, we believe that for this particular case, background noise is the main factor for the quality deterioration.

For Dutch, we analyzed in how far the subtitles of the text can be used for forced alignments. In order to assess the closeness of the subtitles to what is actually spoken, we annotated 52 sentences from a video, and treated the subtitles as hypothesis. The WER is at 26.9% for this segment, while the largest error source are the insertions (18.8%), i.e., the words missing in the subtitles, so that the superfluous speech could be collected by a garbage model. The Dutch ASR performance for this part of the text is at 51.9% due to unoptimized models, and at this stage not usable for our purposes. The next step will be to adopt the models onto the material, and also to see in how far the forced alignment algorithm can cope with the discrepancies of the subtitles with respect to what is actually spoken.

#### 5.2.5 Discussion

Regarding the application scenario in LinkedTV, we conclude that apart from further developing a Berlin dialect model, we need to strengthen our acoustic model for local outdoor interview situations, and we need to strengthen our language model for spontaneous speech.

We have shown that SPSA is an efficient means to optimize free parameters of an ASR decoder. In an unconstrained setting, the WER improves rapidly, but the RTF also increases in an undesirable way. By adding the RTF to the loss function, one is able to stabilize the increase in time requirements. Overall, we have achieved an improvement of 1.6 absolute WER on the DiSCo planned clean task and an improvement of 3.1 absolute WER on the DiSCo spontaneous task, over an already strong baseline.

For future work, we want to work on the following aspects:

- For a very heterogeneous set of free parameters, a linear mapping is somewhat unsatisfactory. SPSA can be extended with an estimate of the second derivate in order to adjust the step size for each parameter given its estimated influence on the overall loss function, which we plan to tackle in a next step.
- The linearly increasing loss function obviously breaks the convergence of the algorithm and can thus should be adopted so that arbitrary RTF convergence criteria can be set. Fixing it is one first

attempt at a solution, but it is quite imprecise. Instead, we plan to employ an adaptive loss function that converges as soon as the given RTF constraint is reached.

- It would be very interesting how SPSA reacts to non-clean speech as especially parameters like insertion penalty could be used as an effective means to set reasonable noise thresholds.

## 5.3 Audio Fingerprinting

### 5.3.1 Problem statement and overview of the State of the Art

With the ever-increasing offer of television content as internet broadcast streams, synchronisation of this material with second screen applications has received considerable interest over the last years, and is also considered to be an interesting technology for LinkedTV as well. We introduce a novel audio fingerprinting method which can be easily implemented, and offer experiments on German news show material. Further, we evaluate possible additional usage for audio fingerprinting in the context of duplicate detection whenever different media shows on the same topic recycle shared media fragments. For a personalized viewer's experience, this knowledge can be used to automatically skip already seen material, recommend similar material or offer more in-depth parts of the program.

While the general topic of media synchronisation has received considerable interest over the last years, some techniques developed there fail to translate to internet streaming applications. Also, some of the existing technologies like [HKO01] or [Wan03] are not easily includable in commercial usage due to intellectual property constraints. The problem of audio fingerprinting itself has found ample attention and the field has found a pretty stable algorithmic basis which is described, for example, in [CBKH05]. Recent developments in this field — like this work — are typically tied to specific applications, especially in the areas of mobile search and social networks, see [CSR11] for a recent review.

### 5.3.2 LinkedTV Approach

The audio fingerprinting algorithm presented here relies on detecting characteristic features in the spectrogram of given recordings. For indexing user-defined segments, the corresponding audio stream is continuously converted into a sequence of spectrogram windows. The frequency range is mapped into 20 roughly logarithmically spaced bins.

For each time position  $t$  (with a step size of 20 ms), and each frequency bin  $b$ , a fingerprint  $f$  is extracted. Each such fingerprint is extracted by comparing the signal's energy at the central point  $(t, b)$  to a set of 16 temporal and frequency-wise neighbors as indicated in Figure 17. This results in a fingerprint represented by a bit string  $f$  of length 16. If the signal in such a neighboring area of  $(t, b)$  has sufficiently higher energy than the central value, the corresponding bit is set to 1, and to 0 otherwise. The fingerprint is kept only if at least one bit is different from zero.

In order to achieve robustness against channel distortions and similar deviations of the signal, each extracted bit sequence is decoded using a linear block code, resulting in a codeword  $c$  for each fingerprint  $f$ . To this end, we employ the binary (24, 12, 8) Golay code allowing to correct up to 3 bit errors in each fingerprint. The 16 bit fingerprints are zero padded to a length of 24 bits to be used together with this code. The resulting codewords  $c$  have a block size of 12 bits. For each time window, a random selection of 500 extracted triples  $(t, b, c)$  consisting of time and frequency information as well as the codeword are stored in an index structure which is then persisted. This results in having several hundred descriptors belonging to a segment of five seconds.

During retrieval time, the complete audio signal is processed in the same way as for indexing. Periodically, the extracted descriptors are used as queries for the index structure, which returns all occurrences of similar descriptors in the indexed material. If the amount of descriptors belonging to a certain indexed segment is sufficiently high, their temporal order is correct and they occur in a comparable duration, a match is declared.

Several measures for detecting distorted version of the original material have been implemented: The use of a linear block code allows for a certain amount of bit errors in the fingerprints, which correspond to energy fluctuations in the underlying signal. By allowing small timing deviations in the order of the spectrogram's Fast Fourier Transform length, different positions of the spectrogram windows are compensated. And finally, the choice of thresholds for the percentage of matched descriptors and their minimal duration allows the adaptation of the algorithm to different use cases with varying requirements regarding precision and recall.

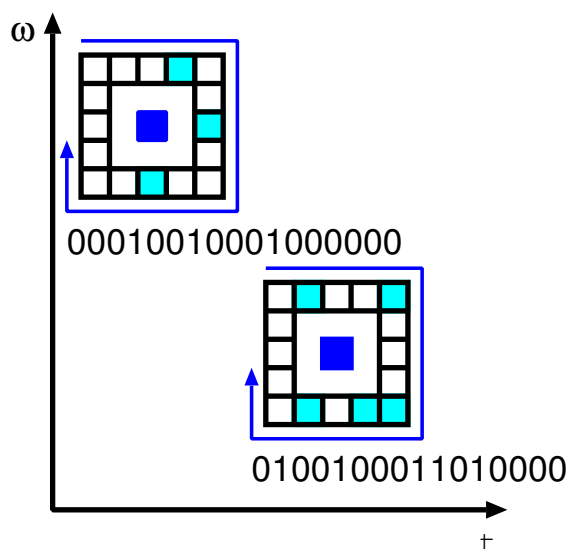


Figure 17: Diagram depicting the audio fingerprint extraction algorithm. The dark blue box in the center represents the central point of the fingerprint.

### 5.3.3 Experiments

First experiments using the fingerprinting algorithm for media synchronisation were carried out using a German TV show. At intervals of ten seconds, one event was indexed, with the goal of detecting these events in potentially distorted material including commercial breaks.

The indexing of 16:30 m of original material with 92 events takes only a couple of seconds on standard PC hardware, the index is smaller than 2 MBytes. The matching runs about 2.5 times faster than real time. Here, the precision of the matching is perfect, i.e., no false positives are detected, while the recall is almost perfect (one miss, i.e., 98.9%). We added white noise, red noise ( $1/f$ ) and brown noise ( $1/f^2$ ) on various signal-to-noise (SNR) ratios to the original signal (Figure 18). While there are still no false positives, the recall drops dramatically at around 20 dB SNR for all three types of noises. As could be expected, brown noise which does not distort as much on the whole frequency range performs best, but only slightly.

Artificial noise is a nice way to test the robustness in general, but it still tells little about actual use case environments. In a second screen application setting, presumably taking place at home, we can expect occasional (possibly loud) noise from sources like banging doors, telephone ringing, and other people talking. For an extreme scenario, we decided that noises that can be heard in a passenger train contain all these sounds, both electronically (announcements, automatic doors, engine sounds) and from persons (other travelers talking, giggling, coughing). Thus, one author recorded these environmental sounds during his daily commute, and we again mixed in these sounds at different SNR ratios (Figure 19). Here, the recall starts to drop only at 10 dB SNR, and still is acceptable at 5 dB SNR.

**5.3.3.1 Duplicate Detection** For preliminary experiments of the fingerprinting technology, we selected seven different news show scenes from the online portal of RBB, on the topic of Berlin's new airport:

- one scene from *RBB um sechs* (2012/08/14), on the delay of its opening,
- two scenes from the *Abendschau* (2012/08/14), one on the delay of its opening, the other on a terrorist warning,
- one scene from *Brandenburg Aktuell* (2012/08/14), on the delay of its opening
- two scenes from *RBB Aktuell* (2012/08/14), one on the delay of its opening, the other on a terrorist warning,
- one scene from *Kontraste* (2012/08/12), on a general progress of the airport.

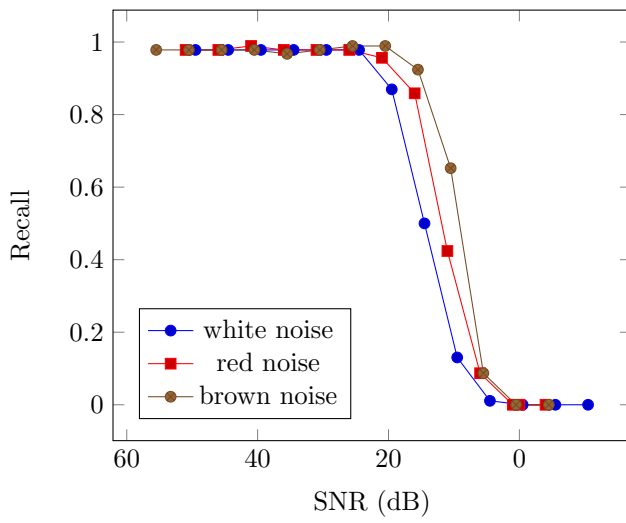


Figure 18: Recall drop with increasing artificial noise level.

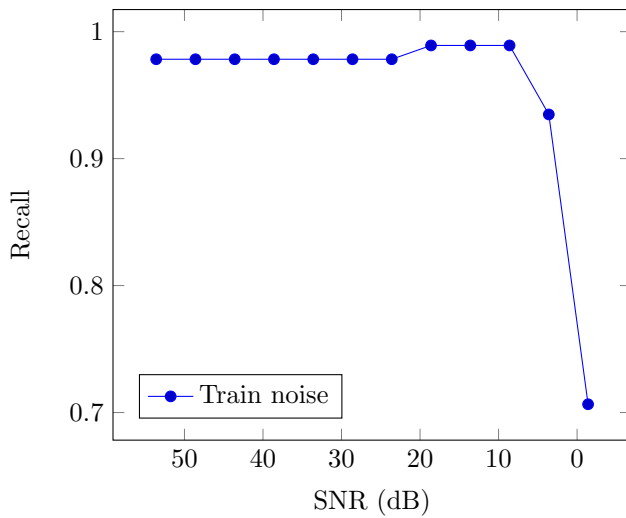


Figure 19: Recall drop with increasing passenger train noise level.

Focusing on the four scenes about the opening delay, we found several interesting differences and commonalities: the host is different for all shows, but the opening material for the report is at times similar. All four reports feature an interview with the politician Anton Hofreiter, but only three of them have the same content; the *RBB um sechs* show takes different quotes from the same material. The show *Brandenburg Aktuell* is the longest, and has a live interview with the politician Jörg Vogelsänger, whereas some shots of Vogelsänger's interview are recycled in *RBB Aktuell*. Both *Abendschau* and *RBB Aktuell* have the same final conclusion drawn by reporter Boris Hermel. See Figure 20 for an overview.

We use the *RBB Aktuell* video as the seed video for the fingerprint algorithm. While the data used is far too small to draw conclusions about the significance, the algorithm was able to detect Hofreiter in the three shows which broadcast the identical utterances, and did not detect a duplicate where other parts of the interview were used. Also, both instances of Vogelsänger and Hermel were detected correctly. As a sanity check, we also ran the fingerprint detection on the other videos which do not contain any duplicates but are on the same topic, and the algorithm correctly reported no matches.

#### 5.3.4 Discussion

We presented a novel audio fingerprinting algorithm. Preliminary experiments indicate that it can robustly synchronize a second screen application with broadcast media content. While established methods with a similar application exist, our method does not have any intellectual property constraints and can be implemented easily, while still providing good results and using reasonable processing power and storage. Further, for the given data we have seen another promising use case, as the technique can also detect duplicates within an internet broadcast archive. This can be used to (a) recommend similar content to the user, (b) allow him to skip already seen material, and (c) indicate whether another show probably offers a more in-depth coverage of, e.g., an interview.

Currently, the timestamp results for the fingerprint detections might be misleading since an identified smaller chunk could also appear at the end of the larger interview. As future work, we want to further strengthen the ties between the videos, by using diarization and speaker detection so that, e.g., the full interview can be linked right from its beginning. On a larger scale, using automatic topic segmentation on the basis of, e.g., shot segmentation, crawled subtitles and/or automatic speech recognition, we can also link to the beginning of larger reports, a feature that might be especially interesting for news summaries at the end of the day where a user wants to hear the full story.




SHOW TIME LENGTH	RBB um sechs 6 pm 1:51 m	Abendschau 7.30 pm 1:54 m	Brandenburg Aktuell 7.30 pm 3:33 m	RBB Aktuell 9.45 pm 2:18 m
				
				
				
				
				
				
				
				

Figure 20: Content of four news shows containing material about Berlin's new airport.

## 6 Keyword Extraction and Named Entity Detection

The motivation for keyword extraction is to provide a lower-dimensional digest on the main topics of the input video.

Perhaps the most well known methods used in natural language processing for providing “digests” of the information in an input document are document summarization and relation and keyword extraction.

Document summarization methods are employed [DM07] to provide shorter summaries of input documents. Document summarization algorithms either select only the important sentences from the input text, or even amalgamate shorter textual fragments to synthesize new sentences. Much of the document summarization output are words with small *information content* that bind together the text and make it intelligible for humans. In order to perform the latter operation, some of the document summarization approaches use relation learning as a component [WC12]. Relation extraction (sometimes also relation learning) [BB07] aims at discovering entities in the input text *and* relations between these entities. In contrast, keyword extraction deals only with discovering the entities.

The motivation to provide a more concise description of video content comes from WP2 and WP4. Starting from WP4, the requirement posed in D4.2 is to obtain machine-readable semantic description of videos that the user has watched. This description is provided using a set of entities, which can be linked to an ontology using the `rdf:type` relation. Any interaction between these entities is not considered. As a consequence, the use of document summarization or relation extraction algorithms in addition to keyword extraction is not necessary.

The fact that the extracted keywords are required by the WP2 and WP4 to be linked to the Linked Open Data cloud imposes additional requirements on the keyword extraction algorithm, which are not commonly found in the literature or software implementations: WP1 provides the recognition of entities in the input text along with their weight. WP2 provides entity recognition, assigning each entity to `rdf:type`, reusing some of the algorithmic results on entity recognition provided by WP1.

The role of WP1 in the LinkedTV data annotation process is to provide means for

- entity recognition,
- weight assignment.

These tasks need to be performed in close collaboration with WP2, which provides the entity classification service. For this reason, the concept of *keyword* is united with the concept of *entity*, which in turn is defined as a noun or proper noun with modifiers.

### 6.1 Keyword recognition

The first phase of the (WP1) keyword extraction or the (WP2) entity classification is the identification of candidate keywords or entities. These are either single words or multi-word phrases (*keyphrases*). Within LinkedTV, several algorithms for keyword recognition were considered.

#### 6.1.1 Initial version: all words

The initial version of the algorithm considered all words in the input text as keywords. An extensive language-specific stop-word list was used to exclude words with low information content.

The LinkedTV-specific drawback of this approach is that some of the recognized candidate keywords are not entities<sup>3</sup>, which cannot be used in the WP2 entity classification process.

Another source of incompatibility with WP2 entity classification tools is the fact that keywords are single words, while WP2 tools generally operate on the level of noun chunks.

The initial version of the algorithm as described here was deployed as the LinkedTV Keyword extraction tool.

#### 6.1.2 Language-specific noun phrase chunking

This subsection describes keyword extraction using a language-specific extraction grammars.

Prior to developing new extraction grammars, we considered using the noun phrase extraction module readily available in the GATE framework. After small-scale experimentation we concluded that this module is not suitable for entity recognition, due to what we have perceived to be results inconsistent

---

<sup>3</sup>Entities are nouns with modifiers.



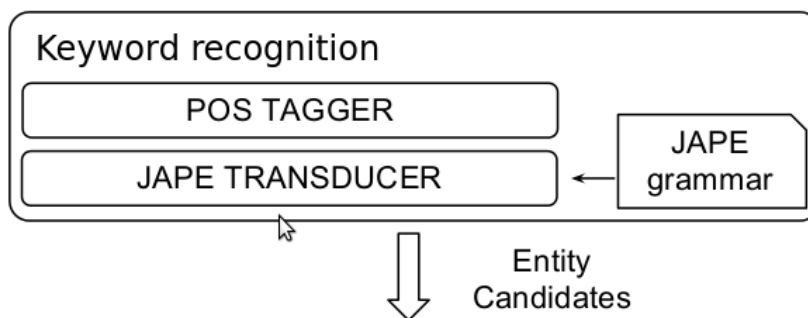


Figure 21: Current LinkedTV keyword recognition workflow.

with our definition of candidate entity. Another issue was that the recognized noun phrases included also the leading determiners, which would have to be removed in an additional processing step. Finally, the GATE noun phrase extraction module is not available for Dutch and German.

Extraction grammars were designed for German and Dutch using the JAPE language [CMT00]. The JAPE language allows to define regular expressions over linguistic annotations. The grammars need to be tied to a specific set of linguistic annotation tags generated by the employed Part-of-Speech (POS) tagger. The workflow is depicted at Figure 21.

Due to the constraints imposed by the set of supported languages, the available options for choosing the POS tagger were significantly reduced. After reviewing the options, the freely available TreeTagger tool<sup>4</sup> was selected. The advantage of this tool is that it supports both German and Dutch, albeit with a different tagset for each of the languages.

The TreeTagger tool is freely available, but for commercial use, the permission of the author needs to be obtained. A free license is granted for evaluation, research and teaching purposes.<sup>5</sup>

In line with the effort to streamline the WP1 and WP2 text processing, the developed entity recognition grammars were incorporated both to the WP1 Keyword Extraction tool and to the WP2 Targeted Hypernym Discovery (THD) entity classification tool. For the latter system, two variants of the grammars for each language were developed. One focusing on what we call *common entities*, the other on *named entities*. Since named entities generally provide higher information content than common entities, this distinction might be used to provide weight for the candidate entity. The result of application of the extraction grammars on a document (an ASR transcript) is a set of *keyphrases* (rather than keywords as in our initial approach).

The algorithm described in this subsection is deployed in the current version of the LinkedTV extraction tool.

### 6.1.3 Statistical recognition with Stanford NER

The two previously described approaches to recognizing entities are based either on a naive take-all approach, or a hand-crafted set of rules. Another possibility is to use a solution based on machine learning, which entails training a system based on a large amount of annotated data.

Our attempt to deliver a solution employing machine learning techniques is based on wrapping the Stanford Named Entity Recognizer system<sup>6</sup>, which is implementation of linear chain Conditional Random Field (CRF) sequence models. The system was trained on the CONLL 2003 dataset<sup>7</sup> (German) and on CONLL 2002 dataset<sup>8</sup>.

The Stanford NER is used as a means to identify candidate entities in the input text for example in the AIDA system [YHB<sup>+</sup>11]. The Stanford NER keyword extraction is currently deployed as part of the LinkedTV Semitags tool, but could be prospectively used to provide input set of keyphrases for the WP1 keyword extraction efforts.

Experimental evaluation of this algorithm is covered in Section 6.3.

<sup>4</sup><http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/>

<sup>5</sup>[www.ims.uni-stuttgart.de/~schmid/Tagger-Licence](http://www.ims.uni-stuttgart.de/~schmid/Tagger-Licence)

<sup>6</sup><http://nlp.stanford.edu/software/CRF-NER.shtml>

<sup>7</sup><http://www.cnts.ua.ac.be/conll2003/ner/>

<sup>8</sup><http://www.clips.ua.ac.be/conll2002/ner/>

Table 11: Keyword extraction precision (p) and recall (r) on the various rbb segments, based on the ASR performance, for keywords with a focus on person names and places.

segment	ASR performance (WER)	hypothesis				reference			
		semantic		entity		semantic		entity	
		p	r	p	r	p	r	p	r
new airport	36.2	0.2	0.6	0.4	0.9	0.2	0.6	0.5	1.0
soccer riot	44.2	0.2	0.9	0.6	1.0	0.2	0.9	0.5	0.9
various other news I	9.5	0.2	0.7	0.3	0.9	0.2	0.7	0.5	1.0
murder case	24.0	0.3	0.6	0.2	0.6	0.3	0.6	0.2	0.6
boxing	50.6	0.2	0.6	0.3	0.4	0.2	0.5	0.3	0.7
various other news II	20.9	0.1	0.4	0.2	0.7	0.1	0.4	0.1	0.6
rbb challenge	39.1	0.2	0.6	0	0.5	0.3	0.7	0.1	0.6
weather report	46.7	0.1	0.3	0.1	0.6	0.1	0.4	0.2	0.6

## 6.2 Keyword extraction

The keyword extraction process entails scoring the candidate keywords identified in the entity recognition phase. For scoring, the common TF-IDF algorithm, which was described in D1.1, is used.

The results of the keyword extraction largely depend on the size and quality of the input corpus used to calculate the Inverse Document Frequency (IDF) scores. These are then combined with the number of occurrences of the keyword in the current document to determine the overall importance of the individual keyword. The candidate keywords are sorted according to the weight and the top  $n$  ones are output.

The same algorithm applies when the result of keyword recognition process are keyphrases.

## 6.3 Experimental results

Using our tool, we extracted all possible keywords from 8 news show stories and ranked them based on the confidence. Also, for each story we manually annotated the ten most important keywords for two different sets: (a) with a focus on semantic content, denoted *semantic* in the experiment, and (b) with a focus on names and places, denoted *entity* in the experiment. For entity recognition, the SemiTags tool (refer to Section 6.1.3) is used.

Then, we compute the precision of the extraction method by counting the matches of the ten highest scoring keywords with our ground truth, and we compute the recall by counting all manual keywords that appear somewhere in the automatically generated list, i.e., not necessarily in the first ten entries. Both keywords and named entity recognition depend heavily on the quality of the ASR when no subtitles are present. Thus, we compare the keyword extraction on both ASR output (*hypothesis*) and the reference transcription. See Table 11 for an overview of the results. It turns out that a high ASR quality is not as crucial as we initially thought. The reason for this probably is that the ten-best keyphrase tend to be very specific, rather long (often containing multiple words) and probably occurring more than once.

We also conclude that the algorithm currently performs better for names and places over semantic context based on its score, since the precision for entity-based keywords is usually higher. This is probably due to the fact that names naturally tend to be important descriptors of a text, whereas by general keywords even for human annotators can be often difficult to agree on fixed set of important keywords. The results can be improved either by taking into account the preferences of a particular user or by taking into account the topic of the keyword. A more detailed analysis of further plans in this direction is presented in Section 6.4.

## 6.4 Extensions

The general keyword extraction workflow described above is applicable in the LinkedTV process. However, there are two additional Linked-TV specific factors that can be worked into the keyword extraction algorithms:

- For building the user profile, the entity processing algorithm in WP4 prefers lower granularity keywords that can be linked with some topical domain.

- The documents in the collection can be divided into two groups: a smaller amount of documents the user has in her history and the remaining documents corresponding to ASRs of videos the user has not yet seen.

In the following subsections, these extensions are discussed in detail.

#### 6.4.1 Identifying topical keywords

Currently, we are considering an algorithm that will identify topical keywords. The LinkedTV workflow allows to compute the similarity of keywords by performing “semantic expansion” - the keywords are represented as weight vectors, rather than atomic strings.

The semantic expansion of keywords is made possible by the fact that within LinkedTV, we aim at convergence of keywords and (linked) entities. The main semantic base used by the LinkedTV entity classification tools is DBpedia. Some of these tools (DBpedia Spotlight and the consortium-developed THD) provide a DBpedia-based disambiguation for input entities. Additionally, the assigned class can also be typically mapped to a DBpedia resource, for example using the NERD ontology<sup>9</sup>. DBpedia resources are resolvable to a Wikipedia page, which in turn can be used to obtain a Bag-Of-Words (BOW) representation of the original entity or its type. In contrast to the original extracted keywords, which are currently handled as “atomic” and semantically incomparable, the linked entities (further *linked keywords*) have a robust BOW representation, which allows to compute distance between individual keywords [Kli10].

The first step in identifying keyword topicality is clustering of the linked keywords, e.g. using the K-means algorithm. The distance between two keywords is computed using the Euclidean distance or Cosine similarity from their BOW vectors. The resulting clusters correspond to latent topics in the documents, with individual clusters grouping for example candidate keywords (keyphrases) related to locations, sport, culture or politics. This clustering is performed as a preprocessing step on the entire collection of documents (ASR transcripts). Finally, the topicality of a candidate keyword is obtained by computing the similarity of the BOW representation of the corresponding linked keyword with the BOW representation of the source document.

The proposed algorithm for identifying topical keywords has following steps:

1. Clustering is performed on all linked candidate keywords in the document collection,
2. Each document is assigned to one of the clusters (e.g. with latent topic “politics”),
3. *Topical confidence* for each candidate keyword is computed based on the keyword’s similarity with the centroid of the cluster its *source document* was assigned to in step 2,
4. The topical confidence is combined with the TF-IDF weight to provide the final keyword ranking.

The last step, which entails incorporating the IDF weight (measure of a rarity of the word) into the overall score, ensures that keywords which are topical, but omnipresent, will not be scored highly. An example of such a keyword is “politician” in the “politics” cluster.

#### 6.4.2 Personalized keywords

The availability of the user history can be utilized to provide *personalized keywords*. With the knowledge of the documents the user has seen, the procedure described in Section 6.4.1 can be modified in the following way:

1. Clustering is performed only on the content in user’s history,
2. The candidate keywords are ranked according to the similarity with the centroid of the cluster the keyword is assigned to in step 1,
3. The topical confidence is combined with the TF-IDF weight to provide the final keyword ranking.

The rationale behind this modification is that the weight of the candidate keyword should rise with its relatedness to any of the topics the user is interested in, not necessarily to the topic of the document the candidate keyword comes from.

---

<sup>9</sup><http://nerd.eurecom.fr/ontology>

While personalized keywords are an appealing idea, their implementation within LinkedTV would have severe implications for the established workflow, as WP1 has no knowledge of the user history. Computation of personalized keywords would require WP4 to call the WP1 Keyword extraction module for each particular user; passing the information on the content in the user's history. The second caveat is that the IDF measure requires a considerable number of documents to stabilize, which makes the algorithm applicable only for users with a sufficiently long history of viewed content. This issue can be alleviated by using the IDF computed from all documents if the user's history is too short. The concept of personalized keywords can be considered as a prospective extension of the LinkedTV architecture, but due to the aforementioned concerns it is out of the scope of the LinkedTV project.

## 7 Video Event Detection

### 7.1 Problem statement and overview of the State of the Art

High-level video event detection is now widely recognized as an essential step towards large-scale multimedia content analysis, indexing and search [JBCS13]. Hence, the implementation of a technique for efficient detection and association of events to pieces of media, would provide valuable information for the identification of relations and links between them, contributing to the interconnection of the multimedia content that is envisioned within the LinkedTV scope. However, due to the compositional nature of events (i.e., consisting of actions, actors, objects, locations, times and other components with possible relations among them) [Bro05], this task is much more challenging than tasks dealing with the detection of elementary actions in video [TCSU08] or other, mostly static, semantic concepts. To deal with the inherent complexity of high-level events, typically several low-level features are extracted from the video signal in order to provide a more informative event representation. For instance, the authors in [ADM11] exploit a late fusion strategy of three different feature types, namely, static visual (local image features extracted using a dense sampling strategy and the scale invariant feature invariant transform (SIFT)), audio (Mel-frequency cepstral coefficient (MFCC) descriptors) and dynamic visual features (dense trajectories described with the motion boundary histogram (MBH) descriptor). One support vector machine (SVM) is trained for each feature type and each event of the TRECVID 2011 Multimedia Event Detection (MED) dataset [OAJ11], and the weighted sum of the SVM output scores is used to detect the presence of an event in a test video. Similarly, in [Y. 11], a variety of features (Harris-SIFT, Hessian-SIFT, space time interest points-HOG (STIP-HOG), STIP-HOF, dense HOG, MFCC) are extracted, and a Gaussian mixture model (GMM) supervector is constructed for each feature and each video. The derived GMM supervectors are used to train one kernel SVM (KSVM) for each event in the TRECVID 2011 MED dataset, and the weighted average of the KSVM output scores is exploited for event detection.

Recently, some researchers started to exploit semantic model vectors [SNN03] as a feature representation of high-level events, aiming at better event detection performance. The inspiration behind this modeling approach is that high-level events can be better recognized by looking at their constituting semantic entities. For instance, in [GMK11a] a set of pre-trained concept detectors are used for describing the video signal, and discriminant analysis is used to derive the most informative event concepts. These concepts are then used for describing the videos and for learning the target events. In [al.11, MHX<sup>+</sup>12], large sets of low-level video features as well as semantic model vector features are extracted, and different fusion strategies are used to detect the target events. Experimental results in the above works showed that in some cases event detectors trained using the semantic model vector representation outperformed classifiers trained on state-of-the-art low-level feature representations alone [MHX<sup>+</sup>12], and that their combination with low-level features provides small but noticeable performance gains.

A detailed overview of different state-of-the-art techniques that contribute for the detection of events in media is presented in Section 6.2 of D1.1.

### 7.2 LinkedTV approach

In the above works, fusion of different modalities is performed along different feature types in order to improve the detection performance. However, recent works on machine learning have shown that in various learning problems performance gains can also be achieved by combining multiple classifiers trained along different regions of the same feature space [ETP<sup>+</sup>08, GMKS12]. Building on this, for the detection of events from media in LinkedTV, we intent to use a combination of semantic model vectors for video event representation with a new event detection method that exploits a SRECO framework and the loss weighted decoding (LWD) measure [ETP<sup>+</sup>08, EPR10, EPR09] to combine multiple classifiers trained at different regions of the same concept space.

#### 7.2.1 Problem formulation

Our goal is to learn an event detector  $f: \mathcal{X} \rightarrow [0, 1]$  and the respective threshold  $\theta \in [0, 1]$  for providing a hard decision regarding the presence of the target event in the video. For this, a concept-based representation of an annotated video database is used,  $\{(\mathbf{x}^p, y^p) \in \mathcal{X} \times \{-1, 1\}\}$ , where,  $\mathcal{X} \subset [0, 1]^Q$ ,  $\mathbf{x}^p = [x^{p,1}, \dots, x^{p,Q}]^T$  is the model vector representation of the  $p$ -th video in the dataset. I.e.,  $x^{p,\kappa}$  is the degree of confidence (DoC) that the  $\kappa$ -th concept (out of  $Q$  concepts in total) is depicted in the  $p$ -th video, and  $y^p$  is the label of the  $p$ -th video denoting the target event class ( $y^p = 1$ ) or the “rest of the world” class ( $y^p = -1$ ).

## 7.2.2 Video representation

**7.2.2.1 Low-level visual features** For the extraction of low-level visual features, we follow an approach similar to the one described in [MGS<sup>+</sup>12], as explained in the following. The visual stream of a video is decoded and represented using temporal sequences of keyframes extracted from video at fixed intervals, i.e., one keyframe every 6 seconds.

The spatial information within each keyframe image is encoded using a  $1 \times 3$  spatial pyramid decomposition scheme, i.e., the entire image is the pyramid cell at the first level, and three horizontal image bars of equal size are the pyramid cells at the second level [vdSGS10]. For the detection of salient image patches at the pyramid cells we use either a dense sampling strategy or the Harris-Laplace detector. The statistical properties of a local patch are captured using a set of suitable descriptors to derive an 128- or 384-dimensional feature vector depending on the type of the descriptor. Specifically, we utilize the SIFT descriptor as well as two of its color variants, RGB-SIFT and oponentSIFT [vdSGS10]. Subsequently, for each of the aforementioned sampling strategies, descriptor types and pyramid cells, a Bag-of-Words (BoW) model of 1000 visual words is derived using the k-means algorithm and a large set of automatically extracted feature vectors. The assignment of the derived local feature vectors to the codebook words is done using either hard or soft assignment [vGVSG10]. Therefore, in total  $I = 12$  feature extraction procedures are utilized (called hereafter channels [ZMLS07b]), derived from every combination of sampling strategy (2 options), descriptor type (3 options) and assignment technique (2 options) described above. Applying the above procedure, the  $l$ -th keyframe of the  $p$ -th video sequence is represented with a 4000-dimensional BoW feature vector  $\mathbf{z}_i^{p,l}$  in the  $i$ -th channel feature space  $\mathcal{L}_i$ .

**7.2.2.2 From low-level features to model vectors** A set of  $Q \cdot I$  pre-trained concept detectors,  $\mathcal{G} = \{g_{\kappa,i} : \mathcal{L}_i \rightarrow [0, 1] \mid \kappa = 1, \dots, Q, i = 1, \dots, I\}$ , is utilized to provide an intermediate level representation of a video keyframe based on  $Q$  semantic concepts [GMK11a, MHX<sup>+</sup>12]. A *weak* concept detector  $g_{\kappa,i}$  is designed using a linear SVM and a training set of low-level feature vectors referring to the  $i$ -th channel (Section 7.2.2.1) and the  $\kappa$ -th semantic concept. To derive a *strong* concept detector  $g_{\kappa} : \mathcal{L}_1 \times \dots \times \mathcal{L}_I \rightarrow [0, 1]$  for the  $\kappa$ -th semantic concept, the relevant weak concept detectors  $g_{\kappa,i}, i = 1, \dots, I$ , are combined at the score level using the harmonic mean operator. In this way, the  $l$ -th keyframe of the  $p$ -th video in the database is associated with the model vector  $\mathbf{x}^{p,l} = [x^{p,1,l}, \dots, x^{p,Q,l}]$ , where,  $x^{p,\kappa,l}$  is the response of the strong concept detector  $g_{\kappa}$  expressing the DoC that the  $\kappa$ -th concept is depicted in the keyframe. At this point we should note that a model vector can be similarly derived using the set of the  $Q$  weak concept detectors referring to a specific single channel  $i$ .

**7.2.2.3 From frame-level to video-level representation** The procedure described above provides a set of model vectors for each video (i.e., one model vector for each keyframe). In order to derive a model vector representation of the  $p$ -th video, the model vectors of the individual keyframes referring to it are averaged. For instance, when using the strong concept detectors, the model vector referring to the  $p$ -th video is computed using  $\mathbf{x}^p = \sum_{l=1}^{L_p} \mathbf{x}^{p,l}$ , where  $L_p$  is the length of the  $p$ -th video in keyframes.

## 7.2.3 Event detection

Event detectors are learned separately for each event following a target-event versus rest-of-the-world approach. A detector is derived using a splitting algorithm to partition the event class to several subclasses, then learning a number of subclass event detectors, and finally embedding the pool of the trained subclass detectors within a new variant of the ECOC framework [ETP<sup>+</sup>08, EPR10, GMKS12], as explained in the following.

**7.2.3.1 Subclass divisions** An iterative algorithm is applied in order to derive a subclass division of the target event class [GMKS12, GMK11b, GMKS13]. Starting from the initial, one subclass partition  $\mathcal{X}_+^{(1)} = \mathcal{X}_+$ , where  $\mathcal{X}_+$  is the set of the videos that belong to the target event class, at the  $r$ -th iteration the k-means algorithm is used to divide  $\mathcal{X}_+$  to  $r$  subclasses,  $\mathcal{X}_+^{(r)} = \{\mathcal{X}_j^{(r)} \mid j = 1, \dots, r\}$ .

At each iteration the following non-gaussianity measure is computed along the partitions

$$\Phi^{(r)} = \frac{1}{r} \sum_{j=1}^r (\gamma_j + \beta_j), \quad (10)$$

where,  $\gamma_j = \frac{1}{Q} \sum_{\kappa=1}^Q |\gamma_j^\kappa|$ ,  $\beta_j = \frac{1}{Q} \sum_{\kappa=1}^Q |\beta_j^\kappa - 3|$  are estimates of the multivariate standardized skewness and kurtosis of the  $j$ -th subclass respectively. These are based on estimates of their one-dimensional counterparts, which along the  $\kappa$ -th dimension can be calculated using  $\gamma_j^\kappa = (\frac{1}{P_j} \sum_{x^{p,\kappa} \in \mathcal{X}_j^{(r)}} (x^{p,\kappa} - \mu_j^\kappa)^3) / (\sigma_j^\kappa)^3$  and  $\beta_j^\kappa = (\frac{1}{P_j} \sum_{x^{p,\kappa} \in \mathcal{X}_j^{(r)}} (x^{p,\kappa} - \mu_j^\kappa)^4) / (\sigma_j^\kappa)^4$  respectively. In the above equations  $P_j$  is the number of videos of the  $j$ -th subclass,  $x^{p,\kappa}$  is the  $\kappa$ -th element of the  $p$ -th model vector belonging to the  $j$ -th subclass, and  $\mu_j^\kappa, \sigma_j^\kappa$ , are the sample mean and standard deviation of the  $j$ -th subclass along the  $\kappa$ -th dimension, respectively.

At the end of this iterative algorithm, the best subclass partition  $\mathcal{X}_+^{(H_1)}$  is selected according to the following rule

$$\mathcal{X}_+^{(H_1)} = \underset{r \in [1, R]}{\operatorname{argmin}} (\Phi^{(r)}), \quad (11)$$

where,  $R$  is the total number of iterations and  $H_1$  is the number of subclasses of the target event class corresponding to the derived optimal subclass partition.

**7.2.3.2 SRECOC framework** The application of the iterative algorithm presented above will provide a subclass division of the overall training dataset  $\mathcal{X} = \{\mathcal{X}_1, \dots, \mathcal{X}_{H_1}, \mathcal{X}_-\}$ , of  $H = H_1 + 1$  total subclasses, where  $\mathcal{X}_-$  is the set of videos that belong to the “rest of the world” class. Thus, the video dataset is described at subclass level,  $\{(\mathbf{x}^p, u^p) \in \mathcal{X} \times \{1, \dots, H_1, -1\}\}$ , where,  $u^p$  is the subclass label of the  $p$ -th video denoting that it belongs to one of the subclasses of the target event class ( $u^p \in [1, H_1]$ ) or to the “rest of the world” class ( $u^p = -1$ ).

The derived subclass division is exploited using a ternary SRECOC framework. In particular, a variant of the one-versus-one subclass strategy is used, where binary problems are defined only for subclasses of different classes, similar to [GMKS12]. During the coding step, a set of binary subclass classifiers  $\mathcal{A} = \{a_j : \mathcal{X} \rightarrow [0, 1] | j = 1, \dots, H_1\}$  are utilized, where, the  $j$ -th detector is trained using as positive samples the model vectors of the  $j$ -th subclass ( $u^p = j$ ) and as negative samples the videos with negative label ( $u^p = -1$ ). In addition to the above set of detectors, a last detector  $a_H$  is trained, using as positive samples all samples of the target event, and as negative the rest of the world event samples. Consequently, a codeword  $\mathbf{m}_k \in \{1, 0, -1\}^{1 \times H}$ ,  $k \in [1, H]$  is designed for each subclass, where the codeword referring to the rest of the world event class is defined as  $\mathbf{m}_H = [-1, -1, \dots, -1]$ . In contrary, the elements of the codewords referring to the target event subclasses receive one of the other two ternary digits, i.e,

$$m_{k,j} = \begin{cases} 1 & \text{if } j = k \text{ or } j = H; \\ 0 & \text{else,} \end{cases} \quad (12)$$

where  $k \in [1, H_1], j \in [1, H]$ . The above codewords are then used as rows of the so-called coding matrix  $\mathbf{M} \in \{1, 0, -1\}^{H \times H}$ .

Moreover, in order to update  $\mathbf{M}$ , following the conventional recoded ECOC (RECOC) [EPR09] and pursuing a Loss-Weighted decoding (LWD) scheme, the weighting matrix  $\tilde{\mathbf{M}} \in \mathbb{R}^{H \times H}$  is calculated using the training set and the derived subclass classifiers [EPR10]. This is done by firstly computing the performance matrix  $\mathbf{B} \in \mathbb{N}^{H \times H}$ , whose element  $b_{k,j}$  corresponds to the performance of  $a_j$  on classifying the training samples belonging to the  $k$ -th subclass

$$b_{k,j} = \frac{1}{P_k} \sum_{p=1}^{P_k} s_{k,j}^p, \quad (13)$$

$$s_{k,j}^p = \begin{cases} 1 & \text{if } a_{k,j}^p \geq \theta_j; \\ 0 & \text{else,} \end{cases} \quad (14)$$

where,  $s_{k,j}^p, a_{k,j}^p$  are the response and DoC of the  $j$ -th indicator function and detector respectively, with respect to the  $p$ -th model vector of the  $k$ -th subclass,  $\theta_j$  is the detection threshold referring to the  $j$ -th detector, and  $P_k$  is the number of videos of  $k$ -th subclass. The weighting matrix is then obtained by normalizing each row  $\mathbf{b}_k$  of  $\mathbf{B}$  to unit  $l1$  norm, i.e.,  $\tilde{m}_{k,j} = b_{k,j} / \|\mathbf{b}_k\|_1$  so that  $\|\tilde{\mathbf{m}}_k\|_1 = 1$ , where  $\|\cdot\|_1$  is the  $l1$  norm function. The above normalization effectively allows the treatment of  $\tilde{\mathbf{M}}$  as a discrete probability density function. Subsequently, a performance threshold  $\varphi \in [0.5, 1]$  is used to update (recode) the positions of  $\mathbf{M}$  coded with zero according to the following rule

$$\check{m}_{k,j} = \begin{cases} 1 & \text{if } \tilde{m}_{k,j} > \varphi \cdot \tilde{m}_{k,k} \ \& \ m_{k,j} = 0 \\ m_{k,j} & \text{else,} \end{cases} \quad (15)$$

where,  $\check{\mathbf{M}}$  is the recoded matrix, and  $k \in [1, H_1], j \in [1, H]$ .

During the decoding stage, a test model vector  $\mathbf{x}^t$  is classified to one of the subclasses by first evaluating the  $H_1$  subclass detectors in order to create a codeword for it, and then comparing the derived codeword with the base codewords in the coding matrix referring only to the target event subclasses. For the comparison of the codewords we use the linear LWD measure considering the intersection of the confidence intervals derived from the subclass classifiers [EPR10]

$$d_k^t = - \sum_{j=1}^{H_1} \check{m}_{k,j} a_j^t \check{m}_{k,j}, k = 1, \dots, H_1, \quad (16)$$

where,  $\check{m}_{k,j}, \check{m}_{k,j}$  are the elements of the recoded and weighting matrix, respectively, that correspond to the  $j$ -th subclass and the detector that separates the  $k$ -th subclass from the “rest of the world” class. Note that  $\check{m}_{k,j} \in \{0, 1\}$ ,  $\check{m}_{k,j}, a_j^t \in [0, 1]$ ,  $\sum_{j=1}^{H_1} \check{m}_{k,j} = 1, \forall k, j$ , and therefore  $d_k^t \in [0, -1]$ . To this end, in order to derive a probability estimate for the  $j$ -th subclass, we negate the LWD distance  $\pi_k^t = -d_k^t$ . Finally, considering that all detectors refer to subclasses of the target event, i.e., they can be considered as expert detectors of the event in a subregion of the concept space, an overall DoC  $f^t$  regarding the presence of the event in the test video is obtained using the sum probability rule under the equal prior assumption along the event subclasses [KHDM98]

$$f^t = \frac{1}{H_1} \sum_{k=1}^{H_1} \pi_k^t. \quad (17)$$

The test video is then classified to the target event according to the rule  $f^t \geq \theta$ , where,  $\theta \in [0, 1]$  is the detection threshold value estimated using a cross-validation procedure.

## 7.3 Experimental evaluation and comparisons

### 7.3.1 Dataset description

For the evaluation of the described algorithm as well as its comparison with the kernel SVM (KSVM) [MHX<sup>+</sup>12, Vap98], we used the video datasets of the TRECVID MED 2010 and 2011 tasks. The former dataset (TRECVID MED 2010) consists of 1745 development and 1742 test videos belonging to one of 3 target events (“assembling a shelter”, “batting a run in” and “making a cake”) or to the “rest of the world” event class. For the annotation of the videos we employ the labeling information provided in [MHX<sup>+</sup>12]. The TRECVID MED 2011 consists of 13,871 development videos, 32,061 test videos and 11 event classes, i.e, the “rest of the world” event class and 10 target event classes: “birthday party”, “changing a vehicle tire”, “flash mob gathering”, “getting a vehicle unstuck”, “grooming an animal”, “making a sandwich”, “parade”, “parkour”, “repairing an appliance”, “working on a sewing project”. On average, around 50 and 130 videos per event of interest are included in the development collection of the TRECVID MED 2010 and MED 2011 dataset, respectively.

### 7.3.2 Evaluation metric

For assessing the performance of the individual target event detectors the average precision (AP) is used. The AP summarizes the shape of the precision recall curve and for the  $n$ -th event it is computed as follows

$$AP_n = \frac{1}{M_n} \sum_{s=1}^S \frac{M_n^s}{s} R_s, \quad (18)$$

where,  $S$  is the total number of test samples,  $M_n$  is the number of samples of the  $n$ -th event in the test set,  $M_n^s$  is the number of samples of the  $n$ -th event in the top  $s$  ranked samples returned by the detection method, and  $R_s$  is an indicator function with  $R_s = 1$  if the  $s$ -th video in the ranked list belongs to the  $n$ -th event and  $R_s = 0$  otherwise. The overall performance of a method along all events in a dataset is measured using the mean average precision (MAP) defined as the mean AP along all the events in the database, i.e.,  $MAP = \sum_{n=1}^N AP_n$ , where  $N$  is the total number of the target events in the dataset.

### 7.3.3 Experimental setup

The TRECVID SIN 2012 dataset is used to derive one weak concept detector for each of the  $Q = 346$  TRECVID SIN 2012 Task concepts and for each of the  $I = 12$  feature extraction procedures. Additionally,



<i>Event</i>	<i>K SVM</i>	<i>SRECOC</i>	<i>% Boost</i>
Assembling a shelter	0.20371	0.20472	0.4%
Batting a run in	0.64855	0.65492	1%
Making a cake	0.28803	0.30448	5.7%
MAP	0.3801	0.38804	2.1%

Table 12: Evaluation performance on the TRECVID MED 2010 dataset using weak concept detectors.

<i>Event</i>	<i>K SVM</i>	<i>SRECOC</i>	<i>% Boost</i>
Assembling a shelter	0.25102	0.26869	7%
Batting a run in	0.74314	0.75356	1.4%
Making a cake	0.20375	0.25396	24.6%
MAP	0.3993	0.4254	6.5%

Table 13: Evaluation performance on the TRECVID MED 2010 dataset using strong concept detectors.

a set of  $Q = 346$  strong concept detectors is also formulated as described in Section 7.2.2.2. Subsequently, following the procedure described in Section 7.2.2, each video in the evaluation set is decoded, and one keyframe every 6 seconds is uniformly selected. A set of 13 model vectors for each keyframe is then retrieved using the 12 weak concept detectors as well as the strong concept detector described above. Finally, the model vectors referring to the same video and the same type of concept detectors are averaged, providing 13 model vectors in  $\mathbb{R}^{346}$  for each video. Then, we form 3 evaluation sets of model vectors:

- 1) TRECVID MED 2010 - weak concept detectors: this set consists of the TRECVID MED 2010 model vectors derived using the weak concept detectors referring to the dense sampling strategy, the oponentSIFT descriptor and the soft assignment BoW technique.
- 2) TRECVID MED 2010 - strong concept detectors: this set consists of the TRECVID MED 2010 model vectors derived using the strong concept detectors.
- 3) TRECVID MED 2011 - weak concept detectors: similarly to the first set, this set consists of the TRECVID MED 2011 model vectors referring to the weak concept detectors created using dense sampling, oponentSIFT and soft assignment of visual words.

Our choice to exploit the weak concept detectors referring to the channel combining dense sampling, oponentSIFT and soft assignment, is based on the recommendation by several researchers that this channel provides the best detection performance (e.g., see [vdSGS10]). Therefore, in particular for the TRECVID MED 2010, we can compare the event detection performance of a method that uses strong concept detectors with the one using the best weak concept detectors.

The event detectors for each method and for each of the 3 evaluation sets described above are then created using the corresponding development set. For the K SVM and the base classifiers of SRECOC we used the K SVM implementation provided in the libsvm package [CL11] with radial basis function (RBF) kernel. During training, we need to estimate the scale parameter  $\sigma$  of the RBF kernel and the penalty term  $C$  of the SVM, while for the SRECOC we additionally require the estimation of the recoding performance threshold  $\varphi$ . Following the recommendation in [JBCS13], we set the scaling parameter  $\sigma$  to the mean of the pairwise distances between the model vectors in the development set. The other two parameters  $C$  and/or  $\varphi$  are estimated through a grid search on a 3-fold cross-validation procedure, where at each fold the development set is split to 70% training set and 30% validation set. The estimated parameters are then applied to the overall development set in order to derive the target event detectors.

### 7.3.4 Results

The performance of the SRECOC and K SVM in terms of AP and MAP on the 3 evaluation sets described above are shown in Tables 12, 13 and 14. From the analysis of the obtained results we observe that in the case of the weak concept detectors, SRECOC provides an approximate boost in performance over K SVM of 2.1% and 10.3% in terms of MAP for the TRECVID MED 2010 and TRECVID MED 2011 dataset respectively; when the strong concept detectors are used, the boost in performance in the TRECVID MED 2010 dataset is increased to 6.5%. The small improvement in TRECVID MED 2010 dataset with weak concept detectors is explained by considering the fact that this dataset is small and noisy (due to the weak concept detectors) and thus the base subclass K SVMs of SRECOC overfit the

<i>Event</i>	<i>KSVM</i>	<i>SRECOG</i>	<i>% Boost</i>
Birthday party	0.02601	0.02967	14.1%
Changing a vehicle tire	0.13865	0.13823	-0.3%
Flash mob gathering	0.26711	0.27328	2.3%
Getting a vehicle unstuck	0.11441	0.12168	6.3%
Grooming an animal	0.02705	0.04902	81%
Making a sandwich	0.05381	0.06525	21.2%
Parade	0.10639	0.11798	10.1%
Parkour	0.09069	0.09565	5.6%
Repairing an appliance	0.16934	0.19155	13.1%
Working on a sewing project	0.071052	0.091846	29.3%
MAP	0.10645	0.11742	10.3%

Table 14: Evaluation performance on the TRECVID MED 2011 dataset using weak concept detectors.

data. Increasing the robustness of the features by applying the strong concept detectors in TRECVID MED 2010, or using the much larger TRECVID MED 2011 development set, a noticeable performance gain is achieved by SRECOG over KSVM.

Another important conclusion is inferred by the comparison of the performance between the strong concept detectors and the weak concept detectors in the TRECVID MED 2010 dataset. In terms of MAP, the strong concept detectors outperform their weak counterpart. However, in the case of the “making a cake” event the weak concept detectors are superior. We attribute this paradox to the fact that the procedure for building strong concept detectors from the weak ones (which is concept-independent; see Section 7.2.2.2) indeed increases the accuracy of concept detectors on average, but does not necessarily do so for every single one of the considered concepts. Therefore, the set of strong concept detectors may include, for specific concepts, detectors that are actually weaker than the corresponding detectors of the weak detector set, and this may affect performance for events that depend a lot on these concept detectors.

Finally, we should also note that a model vector approach in combination with KSVMs (which is the approach that we use as our baseline for comparison) was proposed in [MHX<sup>+</sup>12] and was used for the detection of the 3 events in TRECVID MED 2010 dataset, achieving MAP  $\simeq$  0.4. The attained performance here, exploiting the strong concept detectors in combination with KSVM or SRECOG is equivalent or better, respectively, compared to the performance reported in [MHX<sup>+</sup>12].

## 7.4 Discussion

A method that uses a concept-based representation and exploits an error-correcting output framework for detecting high-level events in video has been implemented and evaluated. The experimental results on the TRECVID MED task datasets verified the effectiveness of the proposed method for event detection in large-scale video collections and showed that it favorably compares to the state of the art KSVM approach [MHX<sup>+</sup>12]. Moreover, the effect of weak and strong concept detectors in the performance of the event detection system was examined, indicating that a concept-dependent method for combining weak concept detectors may be useful for improving event detection. Straightforward extensions of the proposed method include the incorporation of event detectors trained along subclasses of different feature spaces and/or the exploitation of a more suitable weighting scheme for combining the weak concept detectors, as explained above.

## 8 Object Re-detection

### 8.1 Problem statement and overview of the State of the Art

Object re-detection can be interpreted as a particular task of the image matching problem, that aims at finding occurrences of specific objects within a collection of images and videos. An object re-detection algorithm takes an image that depicts an object of interest (also called query image below) and evaluates its similarity with pieces of visual information, typically by means of image matching, trying to find instances of this object in other images or videos. Extending this procedure with an appropriate annotation step that assigns a descriptive label to the searched image would allow for automatic instance-based labeling of the detected occurrences of this object. The latter could efficiently support the vision of interactive television that LinkedTV users will experience, since the association of visual content with labels is an important and pre-requisite step for finding videos or media fragments with related content, and for establishing links between them.

One of the most popular state-of-the-art approaches for the estimation of similarity between pairs of images is based on the extraction and matching of descriptors that represent global (e.g. color, texture) or local (e.g., edges, corners) image features. However, due to possible changes in the illumination and/or the viewing position between the matched pair of images (caused by scale and/or rotation transformations), the use of scale- and rotation-invariant local descriptors has been proven as more efficient for this task. In this case the matching procedure can be seen as a three-step process, where the first and the second step correspond to the feature points detection and description respectively; to this end, any of the techniques described in Section 3.5.2 of D1.1 can be used. The third step is where matching between pairs of descriptors is performed. Aiming at more accurate and robust results many researchers proposed various techniques for discarding erroneous matches and keeping the most appropriate among them. Indicative examples of these approaches are: (a) geometric verification by computing the homography between the pair of images using the RANSAC method [FB81] or other similar approached like the M-estimator, L-estimator, R-estimator [Hub81] and Least Median of Squares (LMedS) [Rou84] just to name a few, (b) symmetry tests between matched pairs of descriptors and (c) distance criteria among couples of descriptors when matching is based on k-nearest neighbor (k-NN) search.

However, major changes in illumination and pose (rotation and scale) between the matched pair of images may lead to significant reduction of the number of matched descriptors and thus detection failure. To tackle this, the method in [YM09] implements a fully affine invariant extension of the SIFT descriptor (called Affine-SIFT, ASIFT) by simulating the scale and the camera axis parameters, named latitude and longitude angles. Similarly, in [YHCT12] a view- and illumination-invariant image matching technique is described, that defines a valid range of angle and illumination and iteratively estimates the relation between the matched pair of images. Nevertheless, these repetitive tests are time-consuming, and thus inappropriate for real-time operation. To this end, Ta et. al. [TCGP09] proposed an efficient algorithm called SURFTrac, which combines SURF descriptors and motion information in order to predict the position of the interesting points at the subsequent frame, aiming at the restriction of the search area and the reduction of the computation time. Moreover, the development of novel GPU-based implementations of widely used local descriptors (e.g. SIFT and SURF) could speed-up some parts of the matching procedure and improve significantly the overall time efficiency. Indicative GPU-based implementations of these descriptors have been introduced in [HMS<sup>+</sup>07] and [CVG08] respectively.

However, other state-of-the-art approaches from the relevant literature perform object detection without the use of local descriptors. Some of these techniques include a prior segmentation / binarization step [Sib11], while other methods perform rotation and scaling invariant template matching by applying circular and scalar projections, like the Color-Ciratefi algorithm [AK10]. Moreover, some researchers address the image matching problem as a graph matching problem [DBKP11], while a different approach for real-time object detection that is based on a prior learning step has been described in [HKN<sup>+</sup>09]. Based on previous work (see [HBN<sup>+</sup>08]) the authors propose a fast learning method that builds a database of image patches and their mean appearances, which correspond to a range of possible camera viewpoints. For this purpose they employ an approach that is related to geometric blur [BM01]. At the following detection step they match the incoming feature points extracted by the tested image against the calculated mean appearances, resulting in a first estimation of the viewpoint. The latter is further rectified and the final matched pairs of feature points lead to the detection of the searched object. In [HLI<sup>+</sup>10] the authors introduced another learning-based technique for the detection of texture-less objects, based on the calculation of local dominant orientations.

For more information about techniques for image matching, we refer the reader to the Section 6.1 of D1.1, while a review of the state-of-the-art GPU-based implementations the relate to visual content representation is presented in Section 4.3 of the same deliverable.

## 8.2 LinkedTV approach

A semi-automatic approach has been developed for the re-detection of objects of interest in videos from the LinkedTV content. According to this technique the user (i.e. the editor of the multimedia content) initially specifies an object of interest that appears in a video, by demarcating it with a bounding box on a video frame. At this time the user could annotate the selected object with an appropriate label, providing a piece of information about it. Then by running the object re-detection algorithm for the selected object, additional instances of it in subsequent or non-subsequent frames of the video will be automatically detected via image matching and will be highlighted with a bounding box. If a label has been assigned to the searched object at the selection step, this label will be associated to the detected occurrences, thus performing instance-based labeling of the video content.

The initial technique for object re-detection was a baseline OpenCV implementation and its overall work-flow is depicted in Figure 22. The manually selected object of interest and the tested video file are given as input to the algorithm and the first one is matched successively against all the video frames. To this end, the algorithm initially performs feature detection and description to the query image using the SURF algorithm [BETVG08], and the same procedure is then applied to the tested video frame. A brute-force matching algorithm is used for matching pairs of descriptors, where each descriptor from the query image is matched against all descriptors from the tested frame and vice-versa, and the best match occurs from nearest neighbor search (k-NN, for  $k = 1$ ). After this step, a filtering process undertakes to clear out the erroneous matches by employing a symmetry test, where a pair of descriptors is kept if it has been computed in both phases of the bi-directional comparison between the searched image and the video frame. The remaining outliers are discarded by applying geometric constraints that estimate the homography between the pair of tested images using the RANSAC algorithm [FB81]. If these criteria are satisfied by a sufficient number of pairs of descriptors, the object is detected in the video frame and it is then demarcated appropriately with a bounding box, otherwise the object is not appear in the tested frame.

This technique has been tested on some manually selected images and videos from the documentary scenario which is more suitable for this purpose, and provided good results. However, overall processing of the video frames needed a lot of time, making this method inappropriate for real-time instance based-labeling of the media content. More details about the algorithm's performance are provided in the following section.

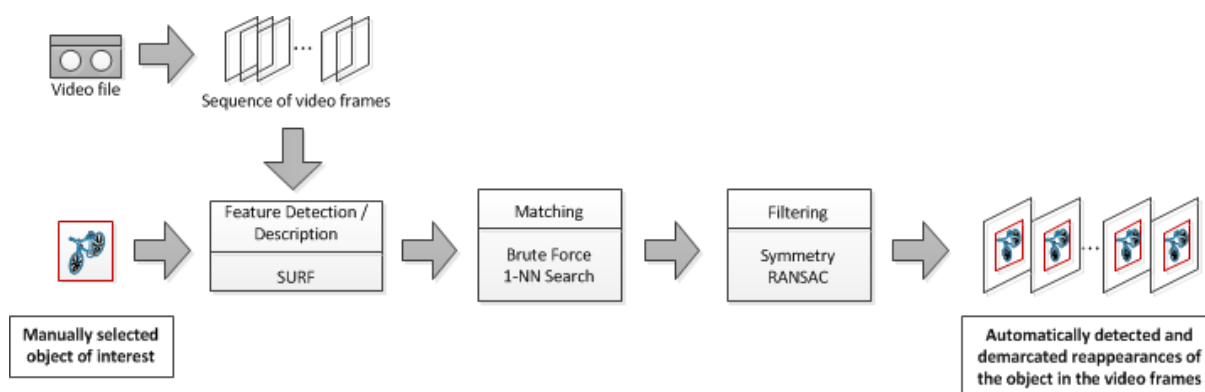


Figure 22: The overall scheme of the initial implementation for object re-detection.

Aiming at further improvement of the object re-detection algorithm's performance, both in terms of detection accuracy and time efficiency, we designed a new method that is comprised by four components: (a) GPU-based processing, (b) artificially generated scale projections of the searched image, (c) efficient filtering of erroneous matches and (d) efficient sampling of video frames. The work-flow of this new version of the object re-detection algorithm is illustrated in the following Figure 23. Again, the algorithm takes as input the manually selected object of interest and the video that has to be processed. Using the first one, the algorithm initially generates a zoomed-in and a zoomed-out version of the object

of interest that will be used for its detection when the latter appears under respective viewing conditions. Then, taking into account the analysis results of the previously described shot segmentation technique for this video (see Section 2.2), the algorithm tries to match the object of interest with the automatically extracted key-frames of each detected shot of the video. If the matching fails, the algorithm checks the generated versions of the object, firstly using the zoomed-out and then the zoomed-in one, and if none of them matches the key-frames of the current shot the algorithm continues with the key-frames of the next one. Differently, if one of the three versions of the object of interest (i.e., the originally selected by the user and the two zoomed artificially generated versions) is matched successfully with at least one of the examined key-frames, then the algorithm continues by matching the object against all the frames of the corresponding shot, using the different versions of the object of interest in the same order as before. After testing the last shot of the video, the algorithm applies an efficient filtering on the detection results, leading to the minimization of the erroneous detections (or misses) that have been occurred from the re-detection procedure. This filtering is based on a sliding window and a set of temporal rules that indicate the existence or absence of the object of interest in the middle frame of this window.

For matching pairs of images the new algorithm follows a similar approach with the previous one, however some steps of this new version have been accelerated by GPU or have been slightly modified in order to achieve the goals for improved detection accuracy and time efficiency (see Figure 23). Both the selected object of interest and the video frames are handled by the GPU. More specifically the feature detection and description as well as the matching part are performed by employing the GPU-based implementations of the SURF and the brute-force matching algorithm, that are provided in the last versions of OpenCV library. However in contrary to the previous version (a) the matching is performed only in one direction i.e. from the searched object to the video frame and not vice-versa and (b) instead of looking for the nearest neighbor, the brute-force matcher is searching for the 2 closest ones (k-NN, for k = 2). Based on the latter the following filtering step, which is performed by the CPU, filters out the outliers by applying a different and more time-efficient criterion based on the distances of each couple of neighbors. According to this, a key-point in the searched image is kept if the ratio of distances of the nearest neighbors is equal or less than a predefined threshold (see the following equation). In this case the algorithm keeps the pair of descriptors that correspond to the closest neighbor.

Keep a key-point in first image, if

$$\frac{DistN_1}{DistN_2} < 0.8 \tag{19}$$

where, DistN1 and DistN2 are the distances of the two nearest neighbors.

As before, the second step of the filtering process is performed by computing the homography between the pair of images utilizing the RANSAC algorithm. Again, the number of remaining pairs of descriptors indicates the existence or absence of the object of interest in the tested frame or key-frame of the video. If the object has been detected in a video key-frame the procedure continues by testing the frames of the corresponding shot, as mentioned before, while if the object has been identified in a video frame it is then highlighted with an appropriate bounding box.

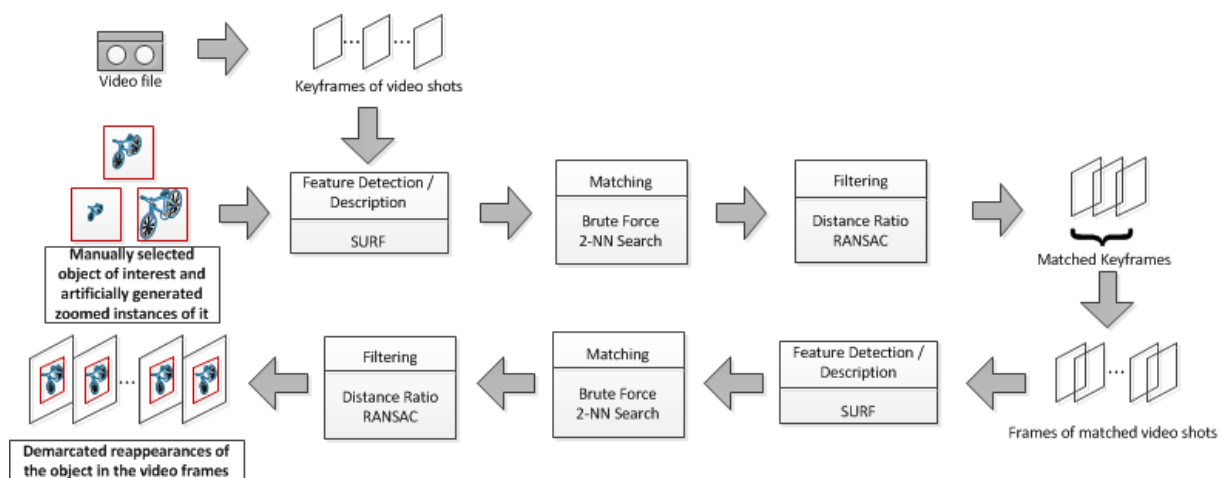


Figure 23: The overall scheme of the improved implementation for object re-detection.

### 8.3 Experimental evaluation and comparisons

The evaluation of the performance of both versions of the object re-detection algorithm was based on some manually selected objects from the documentary scenario of LinkedTV. This scenario includes programs that are dedicated to cultural heritage presenting various objects of art and thus is more appropriate for evaluating the efficiency of these algorithms. For the evaluation of these techniques we considered both the detection accuracy and the time performance.

Our experiments regarding the initial baseline implementation indicate that the algorithm performed quite well, detecting successfully the object for a range of different scales and orientations and for the cases where it was partially visible or partially occluded. Indicative examples of these cases are illustrated in Figure 24, where the query image (presented in top row) is correctly detected after zoom in and zoom out (middle row) and occlusion with or without rotation (bottom row). Moreover, we evaluated the detection accuracy for various sizes of the same object of interest by selecting for example an instance where the object was presented from a short distance (occupying almost all the frame size) or less focused versions of it. By analysing the experimental results of this procedure we concluded that the size of the selected object has an important impact to the algorithm's performance. The selection of a quite focused instance of the object of interest (i.e., big image size) led to successful detection in case of major zoom in, while the algorithm failed to detect the zoomed out occurrences. On the other hand a quite distant version of the object resulted in correct detection in case of zoom out, while the detection of zoomed in appearances was unsuccessful. So, the choice of an average size for the object of interest, based on the closest and the most distant appearances of it provided the best results leading to its successful detection for a wide range of different scales. However, in this case the detection fails when major changes are taking place due to significant change of the visual content within the video frame. Moreover, detection failure was also observed when the objects had more complicated shapes, like e.g., vases, glasses, replicas of guns, toy miniatures, etc. For these objects, a piece of background information was always cropped at the object's selection procedure (see for example Figure 25). Due to this fact, the algorithm exhibited a sensitivity in major changes of viewing angle, since the considerable changes in rotation resulted in significant modification of the background information, and thus detection failure.

Regarding time efficiency, the algorithm processes 5 – 6 frames per second (depending on the image size of the selected object) which means that needs about 4 – 5 seconds for processing one second of a video with 25fps frame rate. It is obvious that such time performance is insufficient for the instance-based labeling of a media content. As we described in the introduction of this section, this procedure will be performed semi-automatically by the video editor and thus it has to be accomplished in time that is at least comparable with the actual duration of the processed media.

After improving the initial version of the object re-detection algorithm by applying the modifications described in the previous section (i.e., acceleration with GPU, generation of multiple views of the searched object, efficient filtering of the results and efficient sampling of the video frames), we evaluated the performance of the new version, using the same dataset.

The detection accuracy of the baseline implementation has been further improved, since the creation of the zoomed in and out versions of the selected object resulted in successful detection for the cases where the previous algorithm had failed. The improvement in the algorithms' performance is presented in Figure 26. The first row of this figure depicts two objects of interest that have been searched by the different versions of the object re-detection algorithm. The results for the first one are shown in the middle row, where the algorithm failed to detect the object after major zoom in, zoom in and occlusion, and zoom out. On the other hand the object has been successfully detected in all these cases by the new version of the algorithm, as presented in the third row of this figure. The detection accuracy of the initial algorithm has been enhanced even more by the additional filtering step. This step performs an efficient refinement of the detection results by discarding almost any of the algorithm's false positives and by calculating appropriate bounding boxes for most of the cases of unsuccessful detection. The only case where the new algorithm fails is for extreme changes in scale. Extreme zoom in (using a factor over 200%) leaves only a very small part of the object appear, modifying significantly the visual content and decreasing the similarity with the query image. On the other hand, extreme zoom out (by applying a factor less than 25%) restricts spatially the object of interest to a small area of the tested image and the number of descriptors extracted from this image area is not sufficient for success matching. Moreover, likewise the initial baseline object re-detection algorithm, extreme changes in rotation led to unsuccessful detection in this case too, due to major changes of the background information. Concerning this problem, a possible solution could be a more sophisticated selection tool that will discard all the unnecessary background information. In this way only the visual information related to the object is going to be used



Figure 24: Object of interest (top row) and detected appearances of it, after zoom in/out (middle row) and occlusion-rotation (bottom row).

in the matching procedure, thus enhancing the detection accuracy.

Concerning time efficiency, the new improved version of the algorithm leads at significant degradation of the processing time, which now ranges between 0.4 – 0.6 of real-time (depending again on the image size of the selected object). This time performance allow us to process a media fragment in times that are smaller than its actual duration, even in the case where all the frames of this fragment have to be checked. However, by improving the algorithm introducing the part that performs efficient sampling of the video frames, we accelerated even more the overall procedure since after this modification the algorithm searches the object of interest only in the frames of specific shots of the video (based on the matching results with the key-frames of each one of them) and not with the entire set of the video frames. The conducted experiments have shown that the re-detection of an object within a media fragment can be accomplished in time that is comparable to the duration of its appearance in this fragment, while the accuracy of the re-detection process remains at high levels.

## 8.4 Discussion

The experimental evaluation of the new version of the object re-detection algorithm showed that we have made a significant progress in comparison with the initial baseline implementation. This progress has been done both in terms of detection accuracy and time efficiency. The new version of the object re-detection algorithm allows for successful detection in cases where the old one failed, while at the same time the processing time has been accelerated about 9 times, making the overall processing time comparable and even less than the actual duration of a media fragment. Moreover the efficient search of specific frames of the media fragment, by exploiting information from the shot segmentation analysis, makes the object re-detection algorithm suitable for real-time use. A video editor can manually select an object of interest and after a reasonable period of time, which can vary from about the half of the duration of the processed media fragment to far smaller times, if the object appears only in some parts of it, all the instances of this object will be automatically detected and highlighted. A future plan at this direction is to decrease the algorithm's sensitivity to the size of the matched object of interest, thus making the user's selection more easy. Moreover, aiming to address the problem of unsuccessful detection of objects with complicated shapes under different viewing angles, we will search on new ways for the efficient selection of these objects, so that the "noise" that background information inserts in the matching procedure will be minimized.





Figure 25: The currently used rectangular selection tool crops a piece of background information, when the object of interest has a more complicated shape, importing “noise” in the matching procedure and resulting in possible detection failure when the object is seen by a different viewing angle.



Figure 26: Two query images (top row) which are not detected by the baseline implementation for major zoom in/occlusion and zoom out (middle row) and their successful detection by the improved version (bottom row).

## 9 Conclusion

This document presents the current state of automatic content analysis tools that will be used in the LinkedTV project. As the goal of these analysis steps is to extract meaningful information that can be used for linking to relevant content, the performance of the currently existing implementations is evaluated on actual data from within the consortium and compared to other approaches wherever possible.

The tools for content analysis include methods for visual, acoustic, and textual data. In the visual domain first a shot segmentation (Section 2) is performed. For every shot first a face detection (Section 3) is performed, indicating whether a human face is visible in the corresponding keyframes. Once faces are found, they can either be clustered (yielding information about re-occurring faces in a shot or a video) or used for face recognition (yielding unique labels for faces that were previously added to a database of people). In addition to working on faces, video concept detection (see Section 4) is used to identify high-level concepts that are present in a given video recording. These concepts, like “landscape”, “throwing” or “press conference”, allow users to scan archives for relevant content and also allow linking additional resources to single shots. Video event detection (Section 7) produces conceptually similar results, but



uses another approach and focuses on the temporal sequence that is inherent in events like “birthday party” or “repairing an appliance”. The last image-based approach is object re-detection, as presented in Section 8. The goal is to automatically find objects of interest by running a query-by-example keyframe retrieval algorithm on user-defined regions in an image.

In the acoustic domain, three fundamental approaches are being pursued. Speaker identification is the acoustic equivalent to face recognition and yields unique identifiers for speakers that are present in a database. Automatic speech recognition (ASR) converts speech to text and therefore allows subsequent textual analysis of spoken content. In order to adapt the current systems for Dutch and German ASR to the usage scenarios several approaches are being evaluated (see Sections 5.2.2 and 5.2.3). Additionally audio fingerprinting (see Section 5.3) is used to detect duplicates in an archive, introduced by broadcasting e.g. news items several times.

Finally in the textual domain, keyword recognition (Section 6.1) and keyword extraction (Section 6.2) are used to identify the most relevant words or phrases in textual content. These can subsequently be used to link to other resources or videos.

So in conclusion WP1 provides already with this first release a broad range of techniques for analyzing the various aspects of multimedia content and ensures that the subsequent automatic and semi-automatic interlinking process has rich meta-data to work with. As the work continues, fine-tuning the approaches to the application scenarios will enhance the quality of analysis results and advance the state-of-the-art in several domains.

## Bibliography

- [ADM11] M. Ayari, J. Delhumeau, and M. Douze et al. INRIA at TRECVID'2011: Copy detection & multimedia event detection. In *Proc. TRECVID 2011 Workshop*, Gaithersburg, MD, USA, December 2011.
- [AK10] S. A. Araújo and H. Y. Kim. Color-Ciratefi: A color-based RST-invariant template matching algorithm. *17th Int. Conf. Systems, Signals and Image Processing*, 2010.
- [AKC<sup>+</sup>02] Walter D Andrews, Mary A Kohler, Joseph P Campbell, John J Godfrey, and Jaime Hernández-Cordero. Gender-dependent phonetic refraction for speaker recognition. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages 1–149. IEEE, 2002.
- [al.11] P. Natarajan et al. BBN VISER TRECVID 2011 multimedia event detection system. In *Proc. TRECVID 2011 Workshop*, Gaithersburg, MD, USA, December 2011.
- [BAB11] J. Baber, N. Afzulpurkar, and M. Bakhtyar. Video segmentation into scenes using entropy and surf. In *Emerging Technologies (ICET), 2011 7th International Conference on*, pages 1–6, sept. 2011.
- [BB07] Nguyen Bach and Sameer Badaskar. A survey on relation extraction. *Language Technologies Institute, Carnegie Mellon University*, 2007.
- [BETVG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.
- [BHK97] Peter N. Belhumeur, P. Hespanha, and David J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages 711–720, 1997.
- [BM01] Alexander C. Berg and Jitendra Malik. Geometric blur for template matching. In *CVPR (01)*, pages 607–614, 2001.
- [Bro05] N. R. Brown. On the prevalence of event clusters in autobiographical memory. *Social Cognition*, 23(1):35–69, 2005.
- [BSB<sup>+</sup>10] Doris Baum, Daniel Schneider, Rolf Bardeli, Jochen Schwenninger, Barbara Samlowski, Thomas Winkler, and Joachim Khler. DiSCo — A German Evaluation Corpus for Challenging Problems in the Broadcast Domain. In *Proc. Seventh conference on International Language Resources and Evaluation (LREC)*, Valletta, Malta, may 2010.
- [CBKH05] Pedro Cano, Eloi Battle, Ton Kalker, and Jaap Haitsma. A review of audio fingerprinting. *The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology*, 41(3):271–284, November 2005.
- [CDHL11] Modesto Castrillon, Oscar Dniz, Daniel Hernndez, and Javier Lorenzo. A comparison of face and facial feature detectors based on the ViolaJones general object detection framework. *Machine Vision and Applications*, 22:481–494, 2011.
- [CL11] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, May 2011.
- [CMT00] Hamish Cunningham, Diana Maynard, and Valentin Tablan. JAPE - a Java Annotation Patterns Engine (Second edition), Department of Computer Science, University of Sheffield, 2000. Technical report, 2000. Technical Report.
- [CSR11] Vijay Chandrasekhar, Matt Sharifi, and David A. Ross. Survey and evaluation of audio fingerprinting schemes for mobile query-by-example applications. In *ISMIR*, pages 801–806, 2011.

- [CVG08] N. Cornelis and L. Van Gool. Fast scale invariant feature detection and matching on programmable graphics hardware. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPRW '08*, pages 1–8, June 2008.
- [D<sup>+</sup>01] George Doddington et al. Speaker recognition based on idiolectal differences between speakers. In *Proc. Eurospeech*, volume 1, pages 2521–2524, 2001.
- [DBKP11] O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(12):2383–2395, 2011.
- [DM07] Dipanjan Das and André FT Martins. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195, 2007.
- [DPG<sup>+</sup>11] Bertrand Delezoide, Frederic Precioso, Philippe Gosselin, Miriam Redi, Bernard Merialdo, Lionel Granjon, Denis Pellerin, Michele Rombaut, and Herve Jegou et al. Irim at trecvid 2011: Semantic indexing and instance search. In *Proceedings of the 9th TRECVID Workshop*, Gaithersburg, USA, December 2011.
- [DVZP04] C. Doulaverakis, S. Vagionitis, M. Zervakis, and E. Petrakis. Adaptive methods for motion characterization and segmentation of mpeg compressed frame sequences. In Aurlio Campilho and Mohamed Kamel, editors, *Image Analysis and Recognition*, volume 3211 of *Lecture Notes in Computer Science*, pages 310–317. Springer Berlin Heidelberg, 2004.
- [EPR09] S. Escalera, O. Pujol, and P. Radeva. Recoding error-correcting output codes. In *Proc. 8th Int. Workshop on Multiple Classifier Systems*, pages 11–21, Reykjavik, Iceland, June 2009.
- [EPR10] S. Escalera, O. Pujol, and P. Radeva. On the decoding process in ternary error-correcting output codes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(1):120–134, January 2010.
- [ETP<sup>+</sup>08] S. Escalera, D. M. Tax, O. Pujol, P. Radeva, and R. P. Duin. Subclass problem-dependent design for error-correcting output codes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(6):1041–1054, June 2008.
- [FB81] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
- [GMK11a] N. Gkalelis, V. Mezaris, and I. Kompatsiaris. High-level event detection in video exploiting discriminant concepts. In *Proc. 9th Int. Workshop on Content-Based Multimedia Indexing*, pages 85–90, Madrid, Spain, June 2011.
- [GMK11b] N. Gkalelis, V. Mezaris, and I. Kompatsiaris. Mixture subclass discriminant analysis. *IEEE Signal Process. Lett.*, 18(5):319–332, May 2011.
- [GMKS12] N. Gkalelis, V. Mezaris, I. Kompatsiaris, and T. Stathaki. Linear subclass support vector machines. *IEEE Signal Process. Lett.*, 19(9):575–578, September 2012.
- [GMKS13] N. Gkalelis, V. Mezaris, I. Kompatsiaris, and T. Stathaki. Mixture subclass discriminant analysis link to restricted gaussian model and other generalizations. *IEEE Trans. Neural Netw. Learn. Syst.*, 24(1):8–21, January 2013.
- [GVSG10] J. Gemert, C.J. Veenman, A. Smeulders, and J. Geusebroek. Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1271–1283, 2010.
- [HBN<sup>+</sup>08] S. Hinterstoisser, S. Benhimane, N. Navab, P. Fua, and V. Lepetit. Online learning of patch perspective rectification for efficient object detection. 2008.
- [HKN<sup>+</sup>09] S. Hinterstoisser, O. Kutter, N. Navab, P. Fua, and V. Lepetit. Real-time learning of accurate patch rectification. 2009.
- [HKO01] J. Haitsma, T. Kalker, and J. Oostveen. Robust audio hashing for content identification. In *International Workshop on Content-Based Multimedia Indexing*, pages 117–124, 2001.
- [HLI<sup>+</sup>10] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab. Dominant orientation templates for real-time detection of texture-less objects. 2010.

- [HMS<sup>+</sup>07] S. Heymann, K. Müller, A. Smolic, B. Fröhlich, and T. Wiegand. Scientific Commons: SIFT implementation and optimization for general-purpose GPU, 2007.
- [HRBO11] Michal Hradis, Ivo Reznicek, Kamil Behun, and Lubomir Otrusina. Brno university of technology at trecvid 2011 sin, ccd. In *Proceedings of the 9th TRECVID Workshop*, Gaithersburg, USA, December 2011.
- [HS88] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of 4th Alvey Vision Conference*, pages 147–151, 1988.
- [HS95] A. Hauptmann and M. Smith. Text, speech, and vision for video segmentation: The infomedia project. In *AAAI Fall Symposium, Computational Models for Integrating Language and Vision*, 1995.
- [Hub81] P. J. Huber. *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley-Interscience, 1981.
- [JBCS13] Y.-G. Jiang, S. Bhattacharya, S.-F. Chang, and M. Shah. High-level event recognition in unconstrained videos. *Int. J. Multimed. Info. Retr.*, to appear in Jan. 2013.
- [JL12] W. Jiang and A. Loui. Video concept detection by audio-visual grouplets. *International Journal of Multimedia Information Retrieval*, 1(4):223–238, 2012.
- [JLM10] Vidit Jain and Erik Learned-Miller. FDDB: A Benchmark for Face Detection in Unconstrained Settings. Technical Report UM-CS-2010-009, University of Massachusetts, Amherst, 2010.
- [JNY07] Y.G. Jiang, C.W. Ngo, and J. Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In *Proc. of the 6th ACM international conference on Image and video retrieval*, pages 494–501, 2007.
- [Ken05] Patrick Kenny. Joint factor analysis of speaker and session variability: Theory and algorithms. *CRIM, Montreal, (Report) CRIM-06/08-13*, 2005.
- [KHB<sup>+</sup>07] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Annual meeting-association for computational linguistics*, volume 45, page 2, 2007.
- [KHDM98] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(3):226–239, March 1998.
- [KL10] Tomi Kinnunen and Haizhou Li. An overview of text-independent speaker recognition: From features to supervectors. *Speech Communication*, 52(1):12–40, 2010.
- [Kli10] Tomáš Kliegr. Entity classification by bag of Wikipedia articles. In *Proceedings of the 3rd workshop on Ph.D. students in information and knowledge management, PIKM '10*, pages 67–74, New York, NY, USA, 2010. ACM.
- [KMSZ10] Alexander Klaser, Marcin Marszałek, Cordelia Schmid, and Andrew Zisserman. Human Focused Action Localization in Video. In *International Workshop on Sign, Gesture, and Activity (SGA) in Conjunction with ECCV*, Hersonissos, Heraklion, Crete, Grèce, September 2010.
- [KP02] Dietrich Klakow and Jochen Peters. Testing the correlation of word error rate and perplexity. *Speech Communication*, 38(1–2):19–28, 2002.
- [LB06] P. Lambert and R.E. Banchs. Tuning machine translation parameters with SPSA. In *Proc. IWSLT*, pages 190–196, 2006.
- [Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707–710, February 1966.
- [LKF09] G. Leon, H. Kalva, and B. Furht. Video identification using video tomography. In *Proc. of IEEE international conference on Multimedia and Expo (ICME)*, pages 1030–1033, 2009.

- [LKS01] A. Lee, T. Kawahara, and K. Shikano. Julius – an Open Source Real-Time Large Vocabulary Recognition Engine. In *Proceedings of Eurospeech*, pages 1691–1694, Aalborg, Denmark, 2001.
- [LL10] Liu Liu and Jian-Xun Li. A novel shot segmentation algorithm based on motion edge feature. In *Photonics and Optoelectronic (SOPO), 2010 Symposium on*, pages 1 –5, june 2010.
- [LM02] R. Lienhart and J. Maydt. An extended set of Haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–900 – I–903 vol.1, 2002.
- [Low04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. of Comput. Vision*, 60:91–110, 2004.
- [LSP06] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. of IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178, 2006.
- [LYHZ08] Xue Ling, Ouyang Yuanxin, Li Huan, and Xiong Zhang. A method for fast shot boundary detection based on svm. In *Image and Signal Processing, 2008. CISP '08. Congress on*, volume 2, pages 445 –449, may 2008.
- [LZZ08] Shouqun Liu, Ming Zhu, and Quan Zheng. Video shot boundary detection with local feature post refinement. In *Signal Processing, 2008. ICSP 2008. 9th International Conference on*, pages 1548 –1551, oct. 2008.
- [MB02] Johnny Mariéthoz and Samy Bengio. A comparative study of adaptation methods for speaker verification. In *Proc. ICSLP*, pages 581–584, 2002.
- [MBM08] S. Maji, A.C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.
- [MDK<sup>+</sup>97] A. F. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET Curve in Assessment of Detection Task Performance. In *Proc. Eurospeech*, volume 4, pages 1899–1903, Rhodes, Greece, September 1997.
- [MGS<sup>+</sup>12] A. Moutzidou, N. Gkalelis, P. Sidiropoulos, M. Dimopoulos, S. Nikolopoulos, S. Vrochidis, V. Mezaris, and I. Kompatsiaris. ITI-CERTH participation to TRECVID 2012. In *Proc. TRECVID 2012 Workshop*. Gaithersburg, MD, USA, November 2012.
- [MHX<sup>+</sup>12] M. Merler, B. Huang, L. Xie, G. Hua, and A. Natsev. Semantic model vectors for complex video event recognition. *IEEE Trans. Multimedia*, 14(1):88–101, February 2012.
- [MKNR12] H.-S. Min, S. Kim, W.D. Neve, and Y.M. Ro. Video copy detection using inclined video tomography and bag-of-visual-words. In *Proc. of the 2012 IEEE International Conference on Multimedia and Expo (ICME)*, pages 562–567, 2012.
- [MMD76] C. Mccamy, H. Marcus, and J. Davidson. A color-rendition chart. *Journal of Applied Photographic Engineering*, 2(3):95–99, 1976.
- [NM65] J.A. Nelder and R. Mead. The Downhill Simplex Method. *Computer Journal*, 7:308, 1965.
- [NRT<sup>+</sup>11] Usman Niaz, Miriam Redi, Claudiu Tanase, Bernard Merialdo, Giovanna Farinella, and Qian Li. Eurecom at trecvid 2011: The light semantic indexing task. In *Proceedings of the 9th TRECVID Workshop*, Gaithersburg, USA, December 2011.
- [OAJ11] P. Over, G. Awad, and J. G. Fiscus et al. TRECVID 2011 - goals, tasks, data, evaluation mechanisms and metrics. In *Proc. TRECVID 2011 Workshop*, Gaithersburg, MD, USA, December 2011.
- [OAM<sup>+</sup>12] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, B. Shaw, W. Kraaij, A.F. Smeaton, and G. Queenot. Trecvid 2012 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2012*, 2012.

- [Och03] F.J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003.
- [OHdJ<sup>+</sup>09] R. J. F. Ordelman, W. F. L. Heeren, F. M. G. de Jong, M. A. H. Huijbregts, and D. Hiemstra. Towards Affordable Disclosure of Spoken Heritage Archives. *Journal of Digital Information*, 10(6), 2009.
- [PC02] Soo-Chang Pei and Yu-Zuong Chou. Effective wipe detection in mpeg compressed video using macro block type information. *Multimedia, IEEE Transactions on*, 4(3):309 – 319, sep 2002.
- [PGB<sup>+</sup>11] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The Kaldi Speech Recognition Toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.
- [PNA<sup>+</sup>03] Barbara Peskin, Jiri Navratil, Joy Abramson, Douglas Jones, David Klusacek, Douglas A Reynolds, and Bing Xiang. Using prosodic and conversational features for high-performance speaker recognition: Report from jhu ws'02. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 4, pages IV–792. IEEE, 2003.
- [PZM96] Greg Pass, Ramin Zabih, and Justin Miller. Comparing images using color coherence vectors. In *Proceedings of the fourth ACM international conference on Multimedia, MULTIMEDIA '96*, pages 65–73, New York, NY, USA, 1996. ACM.
- [QLR<sup>+</sup>09] Zhiyi Qu, Ying Liu, Liping Ren, Yong Chen, and Ruidong Zheng. A method of shot detection based on color and edge features. In *Web Society, 2009. SWS '09. 1st IEEE Symposium on*, pages 1 –4, aug. 2009.
- [RAC<sup>+</sup>03] Douglas Reynolds, Walter Andrews, Joseph Campbell, Jiri Navratil, Barbara Peskin, Andre Adami, Qin Jin, David Klusacek, Joy Abramson, Radu Mihaescu, et al. The supersid project: Exploiting high-level information for high-accuracy speaker recognition. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 4, pages IV–784. IEEE, 2003.
- [Rou84] P. J. Rousseeuw. Least Median of Squares Regression. *Journal of The American Statistical Association*, 79:871–880, 1984.
- [RQD00] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn. Speaker Verification Using Adapted Gaussian Mixture Models. *Digital Signal Processing*, 10:19–41, 2000.
- [SGS10] K.E.A. Sande, T. Gevers, and C.G.M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.
- [Sib11] A. Sibiryakov. Fast and high-performance template matching method. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '11*, pages 1417–1424, Washington, DC, USA, 2011. IEEE Computer Society.
- [SK00] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 746 –751 vol.1, 2000.
- [SNN03] J. Smith, M. Naphade, and A. Natsev. Multimedia semantic indexing using model vectors. In *Proc. Int. Conf. on Multimedia and Expo*, pages 445–448, Baltimore, MD, USA, July 2003.
- [SOK09] A.F. Smeaton, P. Over, and W. Kraaij. High-Level Feature Detection from Video in TRECVID: a 5-Year Retrospective of Achievements. In Ajay Divakaran, editor, *Multimedia Content Analysis, Theory and Applications*, pages 151–174. Springer Verlag, Berlin, 2009.

- [Spa92] James C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37:3, March 1992.
- [Spa98a] James C. Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 34:3, July 1998.
- [Spa98b] J.C. Spall. An overview of the simultaneous perturbation method for efficient optimization. *Johns Hopkins APL Technical Digest*, 19(4):482–492, 1998.
- [SSE08] D. Schneider, J. Schon, and S. Eickeler. Towards Large Scale Vocabulary Independent Spoken Term Detection: Advances in the Fraunhofer IAIS Audiominig System. In *Proc. SIGIR*, Singapore, 2008.
- [SSL<sup>+</sup>11] C.G.M. Snoek, K.E.A. Sande, X. Li, M. Mazloom, Y.-G. Jiang, D.C. Koelma, and A.W.M. Smeulders. The MediaMill TRECVID 2011 semantic video search engine. In *Proceedings of the 9th TRECVID Workshop*, Gaithersburg, USA, December 2011.
- [ST94] Jianbo Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 593–600, jun 1994.
- [TA94] Y. Tonomura and A. Akutsu. Video tomography: An efficient method for camerawork extraction and motion analysis. In *Proc. of Second ACM international conference on Multimedia (ACM MM 1994)*, pages 349–356, 1994.
- [TCGP09] D.-N. Ta, W.-C. Chen, N. Gelfand, and K. Pulli. Surftrac: Efficient tracking and continuous object recognition using local feature descriptors. In *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR '09*, 2009.
- [TCSU08] P. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Trans. Circuits Syst. Video Technol.*, 18(11):1473–1488, November 2008.
- [TKNJ00] Olivier Thyges, Roland Kuhn, Patrick Nguyen, and Jean-Claude Junqua. Speaker identification and verification using eigenvoices. In *Proc. ICSLP*, volume 2, pages 242–245, 2000.
- [TLF08] E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.
- [TMK08] E. Tsamoura, V. Mezaris, and I. Kompatsiaris. Gradual transition detection using color coherence and other criteria in a video shot meta-segmentation framework. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 45–48, oct. 2008.
- [TTZ07] Wenwei Tan, Shaohua Teng, and Wei Zhang. Research on video segmentation via active learning. In *Proceedings of the Fourth International Conference on Image and Graphics, ICIG '07*, pages 395–400, Washington, DC, USA, 2007. IEEE Computer Society.
- [Vap98] V. Vapnik. *Statistical learning theory*. New York: Willey, 1998.
- [vdSGS10] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1582–1596, September 2010.
- [vGVSG10] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek. Visual word ambiguity. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(7):1271–1283, September 2010.
- [VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages 1–511 – 1–518 vol.1, 2001.
- [VSHN12] David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. Jane: an advanced freely available hierarchical machine translation toolkit. *Machine Translation*, 26(3):197–216, September 2012.

- [Wan03] A. Wang. An industrial strength audio search algorithm. In *International Conference on Music Information Retrieval (ISMIR)*, 2003.
- [WC12] Lu Wang and Claire Cardie. Focused meeting summarization via unsupervised relation extraction. *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 304–313, 2012.
- [WLK<sup>+</sup>04] Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, and Joe Woelfel. Sphinx-4: A Flexible Open Source Framework for Speech Recognition. Technical report, Sun Microsystems Inc., 2004.
- [WPZ12] A. Wei, Y. Pei, and H. Zha. Random-sampling-based spatial-temporal feature for consumer video concept classification. In *Proc. of the 2012 IEEE International Conference on Image Processing (ICIP)*, pages 1861–1864, 2012.
- [Y. 11] Y. Kamishima et al. Tokyotech+canon at TRECVID 2011. In *Proc. TRECVID 2011 Workshop*, Gaithersburg, MD, USA, December 2011.
- [YEG<sup>+</sup>06] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Olason, D. Povey, V. Valtchev, and P. C. Woodland. *The HTK Book, version 3.4*. Cambridge University Engineering Department, Cambridge, UK, 2006.
- [YHB<sup>+</sup>11] Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, and Gerhard Weikum. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, 4(12), 2011.
- [YHCT12] Yinan Yu, Kaiqi Huang, Wei Chen, and Tieniu Tan. A novel algorithm for view and illumination invariant image matching. *Trans. Img. Proc.*, 21(1):229–240, January 2012.
- [YKA08] E. Yilmaz, E. Kanoulas, and J.A. Aslam. A simple and efficient sampling method for estimating ap and ndcg. In *Proc. of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 603–610, 2008.
- [YM09] G. Yu and J.-M. Morel. A fully affine invariant image comparison method. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '09*, pages 1597–1600, April 2009.
- [ZMLS07a] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 72(2):213–238, 2007.
- [ZMLS07b] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *Int. J. Comput. Vision*, 73(2):213–238, June 2007.
- [ZMM99] Ramin Zabih, Justin Miller, and Kevin Mai. A feature-based algorithm for detecting and classifying production effects. *Multimedia Syst.*, 7(2):119–128, March 1999.