Master’s Thesis

Interactive Learning for Video Content Analysis

Thesis Committee:
Prof.dr.ir. J. Biemond
Dr. A. Hanjalic
Dr.ir. R.P.W. Duin
Dr.ir. R. Heusdens
Dr.ir. A.R. Bidarra

Author: Ewine A.P. Smits
Email: Ewine@CH.TUDelft.nl
Student number: 1152726
Thesis supervisor: Dr. A. Hanjalic
Date: September 16, 2008 - 10:00
Preface

This thesis is submitted partial fulfilment of the requirements for the degree of Master of Science in Media and Knowledge Engineering. It is preceded by a research assignment under the title “Interactive learning for video content analysis: state of the art”. This document is the final assignment of my studies and reflects the work done within the Information and Communication Theory Group of the EEMCS faculty of the TU Delft, supervised by Dr. Alan Hanjalic during the period September 2007 until September 2008.

Acknowledgements

I am sad that I have finished my time at the TU Delft because it has always been a friendly and challenging environment to study, work and have fun. The last few years I have not only been studying but I have tried to get the most out of my student life by participating actively in the study guild Christiaan Huygens, engaging in promotional activities of the faculty and especially MKE, playing soccer at Ariston’80, being a volunteer bartender in the faculty pub, student assistant, OWEE mentor showing new students around in Delft and EESTEC exchange host. I learned a lot from these activities and met many people along the way and made some life lasting friendships.

I enjoyed working on this project in the department of ICT. The student meetings of the ICT group provided a good place to practice my presentation and discussion skills which turned out to be very useful, in particular for my self confidence.

I would like to thank my parents for always believing in me and supporting me. I would also like to thank Mathijs for his patience, always helping me to relax and stimulate me to keep working. I also want to thank Alan Hanjalic, not only for supervising me during this one year assignment but also making it possible for me to visit the Machine Learning Summer School in Kioloa, Australia, last March, and Leon Rothkrantz for his reference and big help with organizing the funding. It has been a very fun and interesting experience to go to this summer school and attend the lectures from many known and unknown lecturers on many different machine learning topics. It was also a great opportunity to meet other students, PhD students, researchers and professors all interested in the same topics and to talk about their work. Needless to mention I really liked the fact that this summer school was in Australia. I need to thank Umut, who provided and helped me with my data. Last but not least I want to thank my fellow students in the department for the numerous coffee breaks, lunches, interesting on and off topic discussions and for accompanying me to the /Pub.

Ewine Smits
Delft, The Netherlands
September 5, 2008
Thesis summary

We propose a method for identifying segments of a video that represent the events preferred by the user. Possible applications are personalized browsing through music DVDs or smart surveillance systems that can adapt to new circumstances. Requirements for this system are that it is generic and adaptable to the user and to new circumstances. Interactive learning techniques meet these requirements. Interactive learning methods are machine learning methods that involve the user in the learning process in some way. Interactive learning methods do not need information about the problem beforehand and make a system adaptable. For these reasons we propose to apply interactive learning techniques to video content analysis. We chose to implement a user interface that allows users to indicate the relevance of parts of a video by adapting a curve. This curve is used for labelling the underlying audio feature vectors to use as a training set for a classifier.

The scenario in which a user adapts a relevance curve while watching a video turned out to be a good way of interacting with the user. The scenario implicates some restrictions for the system, that have turned out to be very strict for the classifier. This classifier needs to be capable of learning efficiently from different sized training sets, handling unbalanced datasets, handling problems with different complexities and not take too much training and classification time.

The key findings with respect to our music DVD test scenario are:

- Choices about the best classifier and its parameters without any information about events or data can not be optimized because of the lack of prior information.
- Support Vector (SV) classifiers need extensive parameter optimization in order to get good results; no optimization means bad results in general and optimizing based on the (small) training set is very time-consuming and can cause overfitting. Because of the constraints in the SV algorithm, this classifier might not handle highly overlapping classes well. These reasons make a SV classifier not a suitable choice for our application.
- A Nearest Neighbour (NN) classifier, adapted in such a way that the prior probabilities are taken into account, gave the best overall results on our data. This can be explained by the fact that this classifier has no heavy parameter tuning to perform.
- Unbalanced datasets do not have to pose a problem as classifiers can be adapted and post processed to use this information. This same adaptation strategy can be applied for classes that have different classification costs.
- User feedback can be used in an ‘active learning’ manner: Hard to classify samples are labelled and these can be stressed in order to focus more on these samples for better results.

To adapt this to surveillance videos the following steps need to be taken:

- A general video feature set needs to be chosen and, if necessary, combined with the audio feature set.
- Sub sampling needs to be implemented so the system can ‘forget’ old samples and this way adapt to new circumstances.
- The first training is best performed on an offline database or by attaching it to an existing system. This will reduce the training time that is expected to be high because at least a few examples of threat situations need to be labelled, while these situations are rare.

Interactive learning is a promising concept to be applied to video concept detection. This thesis analyses the possibilities and restrictions of applying interactive learning to our scenario.
Table of Contents

PREFACE

ACKNOWLEDGEMENTS

THESIS SUMMARY

CHAPTER 1. INTRODUCTION
  1.1 Research proposal

CHAPTER 2. INTERACTIVE LEARNING

CHAPTER 3. SYSTEM OVERVIEW
  3.1 Online algorithm
  3.1 The user interface
  3.3 Pre-processing steps

CHAPTER 4. FEATURES
  4.1 Feature set 1: short and medium time features
    4.1.1 Energy Level
    4.1.2 Zero Crossing Rate
    4.1.3 Higher order crossing rates
    4.1.4 Level crossing rates
    4.1.5 Pitch
    4.1.6 Band Energy Ratio
    4.1.7 MFCC features
  4.2 Feature set 2: temporal integrated features
  4.3 Feature ranking & selection
    4.3.1 General feature set reduction
    4.3.2 Online feature selection
    4.3.3 Conclusion feature selection

CHAPTER 5. CHOICE OF CLASSIFIER
  5.1 Strengths and weaknesses of the classifiers in theory

5.1.1 Support Vector classifier
Chapter 1.

Introduction

There are a lot of research efforts in the field of multimedia content management concerning the challenges of indexing multimedia and enabling quick, easy and personalized retrieval of multimedia content. Expertises in multimedia signal processing and machine intelligence are combined with state-of-the-art achievements in the fields of "traditional" information retrieval. Bridging the gap between the measurable properties (features) of multimedia signals and the content conveyed by these signals is one of these great challenges. By developing methods for reliably learning user preferences and understanding queries we can try to automatically filter and/or label multimedia content.

The term ‘interactive learning’ will be used throughout this thesis for all the machine learning methods that involve the user in the learning process in some way. This involvement consists of a feedback loop between the user and the system. The information that is exchanged through this feedback channel differs and communication can be initiated by the system and/or by the user. The system can supply the user with useful samples for labelling (active learning) and the feedback of the user on these samples can be integrated in the system in such a way that it iteratively learns from the feedback (relevance feedback). The concept of interactive learning will be explained further in Chapter 2.

Interactive learning is a promising technique that can be applied to fields that classical learning has difficulties with; for instance problems with very little or very expensive training data like medical data or problems that need a flexible, personalized or context dependent solution.

Interactive learning can also be applied to videos. In this system, interactive learning is applied to create a system that is able to alert a user when a part of the video is interesting. Instead of using prior knowledge about events or users, expensive training samples and a priori training, the interaction with the user is used to learn what is relevant to the user and what is not. This information can be collected and used in different ways: using relevance feedback, reinforcement learning, active learning etc. [39]. The user can watch a video and indicate which parts of the video are interesting
and which parts are not. These parts of the video are then converted to training sets by extracting features from the video frames and labelling these sets in 2 classes: relevant and not relevant, using the feedback of the user. The system should generalize this user information and converge as quickly as possible while minimizing user efforts.

A possible application of this system can be smart surveillance systems, both in real-time automatic surveillance systems that raise an alert when necessary and for forensic purposes, like screening security tapes through content-based video retrieval. Existing digital surveillance video applications often need human monitoring for detection of potentially dangerous objects or events. This human monitoring is a very labour-intensive task, and it is generally agreed that it needs a high level of visual attention. The ability to hold attention and to react to rarely occurring events is extremely demanding on a human and prone to error due to lapses in alertness [14]. A smart surveillance system could be very advantageous in aiding human operators in two ways: the ability to generate real-time alerts in the case of a possible threat and the ability to screen security tapes through content-based video retrieval. Quite a few smart surveillance systems are mentioned in literature that do not make use of interactive learning techniques. Currently, these types of systems also work on a dataset that is hard for classical learning methods to train on because of expensive labeling and an unbalanced distribution. This is usually solved by implementing prior knowledge of the environment in the system (what is ‘normal’) and/or constructing models of all possible events that can take place (what situations should cause an ‘alarm’). The current smart surveillance systems are generally very application-dependent like the IBM S3 surveillance system [35] which utilizes threat models in an airport situation and the Video Event Awareness System (VEAS) [12] which constructs and updates a facility model. Other current smart surveillance systems use very high level features like a face detector, object tracking and speech recognition [45]. These non-interactive systems have some disadvantages: they are not general but domain-dependent, they are not able to adapt to new threats because they are static and they need a lot of knowledge and labelled training examples from either the designer or an expert to train the system. Our proposed interactive approach does not need these efforts and knowledge beforehand, but instead it needs an effort of the user, who should be interactively involved in the training of the system by indicating to the system which parts of a video are ‘normal’ and which are ‘alarm’ situations.

There is a lot of research on content based retrieval and on interactive learning, being slightly relevant to this thesis but we have found no related work that is comparable with our interactive video scenario. We will only discuss the work of Meessen et al. [22] that have, very recently, been the first to apply interactive learning to surveillance video retrieval. They propose a system that progressively learns the target scenes thanks to interactive labelling of a few frames by the user without prior knowledge about the events. A set of video surveillance frames are considered that have been pre-processed, including the detection of moving objects as well as the extraction of low-level features describing each object. A retrieval session then consists in an iterative process including querying the user, i.e. to ask him to label a few frames, using this relevance feedback for reweighing the features and inferring a Support Vector classifier from the incremented training set, deducing the class of the unlabelled frames and, eventually, selecting new frames to be presented to the user at the next iteration (an active learning technique). The regularization parameter is optimized on distribution of the training set by a smart
grid search and cross-validation. Correspondences of Meessens system to our scenario are: there is no apriori knowledge available about the events, a Support Vector classifier is used, learning is iterative and interactive, some form of active learning is adopted, the classifier parameters are adapted to the unbalance of the dataset and results are measured by the F5-measure because recall is far more important than precision in a surveillance scenario with rare target events. Besides these correspondences there are also a few differences, the main difference being that the data is already available, pre-processed and features extracted before the interactive process starts. Their active learning approach lets the system query the user to label specifically selected samples while in our proposal this responsibility of picking samples is shifted to the user by correcting the feedback curve. Their experiments show promising results: they demonstrate the efficiency of the approach and show how it allows reaching high retrieval performances. Unfortunately most of these results are based on simulations using a not so realistic percentage of positive samples of 30%.

A different possible application is personalized browsing through or indexing of music DVDs. The current systems try to estimate what music a user likes by using information in the metadata of the music files like artist and album which are often not available or unreliable, or by using information about what other people with a similar taste listen to, for instance the TRIBLER content search and recommendation [31] and the popular online music recommendation system last.fm [50]. Several researches on music recommendation have concluded that collaborative filtering or methods that use musical metadata (genre, artist, etc.) efficiently recommend music. Some of these methods also used content-based information, for instance [5]. But these methods use the Internet and need other users’ likings or annotations for their recommendations and recommend complete albums, artists or songs. To the best of my knowledge there exists no recommender system based on audio features using parts of a song or album or live DVD as entities, although this would be most similar to our interactive system. In our proposed system the user can indicate which parts of a music DVD he likes and dislikes. The system will use these examples to notice the user when an interesting part is playing (or the system could fast-forward to this part). Some irrelevant parts can be skipped, for example all the applauses, or a user can search for instrument solos of different instruments or a specific voice or artist.

The scenario that is used for testing the convergence of our system is based on the application with music DVDs for practical reasons. We will discuss the generalizability of this system in Chapter 8 and the changes that would need to be made to adapt the system to surveillance video applications. We will also look at possible problems that could arise when our solution is adapted to surveillance datasets.
1.1 Research proposal

We propose a method for identifying segments of a video that represent user specified events by using interactive learning with no prior information about the video nor the events. To test this concept also a graphical user interface will be developed. In this interface the system needs to be able to visualize its classification performance while playing the video and the user needs to be able to change this iteratively in an online manner. For testing this interactivity it is necessary for the system to be able to output its classifications along with the video near to real-time. Special research questions in this proposal are:

- What classifier suits this scenario best?
- Is it viable to make choices concerning the classifier, its parameters and other variables for such a general system with no prior information about the video and the events?
- What are the practical differences between this approach and classical, passive learning approaches?
- What are the practical differences between this approach and the classical relevance feedback approach on image collections?
- Will such a system converge well enough and in how many iterations?
- What, if any, adaptations need to be made to this system in order to apply this in practice to surveillance videos?

The development and the evaluation of the system will be performed on a concert database provided by an online concert broadcaster [49], one of the biggest online concert archives in the world containing 900 full-length concerts, festivals, performances, debates and lectures. A variety of the music genres is chosen and annotated for our purposes. The software used for the machine learning tools is a Matlab toolbox named PRTools [8].

The thesis outline is as follows: first we will start with a definition and summary of interactive learning techniques in Chapter 2. In Chapter 3 the system will be discussed in general terms to get an overview of the system in terms of the user interface and interaction and the different components of the system. These components will be discussed in Chapter 4, 5 and 6, in which we will respectively discuss the features that are used, the choice of classifiers and a few post-processing techniques. The results are presented in Chapter 7 and discussed in Chapter 8 that also presents some ideas for future work.
Chapter 2. Interactive learning

Interactive learning will be used in this report as an overall concept title. Other names have included query refinement, interactive retrieval, or emergent semantics [18][32][41]. In this paragraph I introduce some terms and explain the concept interactive learning and the reasons for many researchers to adopt such a strategy in different fields.

First, some of the terminology used in this report is explained. When we consider classification or retrieval, we need input objects that can be classified or retrieved. These input objects or entities can be any multimedia file or part of a multimedia file; this also differs per research proposal. Examples of these entities are:

- images in image retrieval
- complete videos in a personal indexed video archive
- video key frames in a video retrieval system

These classification or retrieval entities will be referred to as samples. When these entities are used for training, we call them training samples. This term will be used for the remainder of this report.

Interactive learning is a form of supervised learning, but it is different from classical supervised machine learning. One main area of supervised learning is the classification task, the task of creating a mapping from input samples to labels, like image topics or {relevant, not relevant}. In classical (passive) learning, a training set must be prepared of manually labelled samples. Next, a learner is used on this training set to generate a mapping from training examples to the labels. This mapping is called a classifier, which can be used to label new, unseen samples. In contrast to classical machine learning, interactive machine learning (IML) methods involve the user in the learning process. In information retrieval this is done by presenting multiple rounds of results for feedback (a result can be relevant or not relevant) so the user can refine the query iteratively. Instead of randomly picking documents to be manually labelled for the training set, objects from the pool that are to be labelled are now more carefully chosen (or queried). Not only will the system gain from the selective training set and feedback, it can also be used to select the important features during the training stage or to adjust the similarity
measure that is used. This means that a large repository of (low level) features can initially be calculated and fed to the learning algorithm so it can learn the best features for the classification problem, instead of letting the user or designer choose the features directly. The user is then focused on rapidly creating training data that will correct the errors of the classifier [10] until the classifier performs in the desired way. The difference between active and passive learning is depicted in the following Figures 2.1 and 2.2 shown below.

![Figure 2.1 Visualization of a passive learner. The training stage and classification stage are completely separate. If new training samples are available to improve the classifier, the complete training process has to start over again.](image1)

In the above figure a passive learner is schematically visualized. The difference to an active learner, visualized in the figure below, is clear: the learner and the user communicate in an iterative way, instead of just supplying training data once and outputting a static classifier.

![Figure 2.2 Visualization of an active learner. The classifier can be renewed every time new training samples are offered to the learner.](image2)

In content-based multimedia retrieval the need for example-based interactive learning became also apparent: it is presented as a possible solution to “bridging the semantic gap” [18]. The semantic gap is the lack of coincidence between the information that one can extract from the multimedia data and the interpretation that a user can give the data in a given situation [38]. This was caused by the lack of understanding of the semantics of the query: the system simply used the underlying low-level features to match one image to another. When the user is involved in the learning process by refining the query iteratively and correcting the errors of the classifier in this way, the learning becomes adapted to the user and his current query. Low-level features are not just used for matching objects using some similarity measure, but the feedback is used to adjust the query, similarity measure or weights of the features, so the system will try to get a better understanding of the semantics of the query in each iteration.

The majority of the classification approaches are passive with the consequence that they can not cover the entire content variety nor be adaptable to user preferences. A summary of the advantages of example-based interactive learning over classical machine learning and multimedia retrieval is given below:
- No need for a large set of manually labelled training data, that is very costly to gather because it involves human intervention for the judging of the “ground truth”
- No need for selecting features prior to the training stage. Many machine-learning algorithms are very sensitive to feature selection and suffer greatly if there are many features. In the interactive machine learning model (IML) the pre-selection of features can be eliminated and transferred to the learning part of the IML if the learning algorithm used performs feature selection
- No time-consuming offline training stage before a working classifier is constructed
- The designer of the software does not need to have expert field knowledge to choose the features or model the desired behaviour
- Interactive learning can cope with the subjectivity of human perception of visual content
- Example-based learning is preferred so that users do not have to try to find the correct words to explain what exactly they are looking for, which is hard (if not impossible) in itself to do, but even harder for a computer to understand the query and translate it to (low-level) features
- The user does not even have to have an exact idea of what he/she is looking for beforehand: an ‘I know it when I see it’ approach is possible
- Interactive learning can help “bridging the semantic gap” as explained above.

Interactive learning has already proven to be very useful in reducing the need for large training sets and in better retrieval results in other areas, from text documents to multimedia retrieval [13][42]. According to Smeulders [38] the information acquired through the interaction patterns with the user contains rich knowledge about the meaning of the pictures and the system should take advantage from this knowledge. The interaction should be an important aspect in any modern image retrieval system, rather than a last resort when the automatic methods fail. Lew et al. [18] also pointed out that one of the major challenges in content-based multimedia information retrieval is interactive search, emergent semantics, or relevance feedback systems. A drawback of interactive learning is that some choices, e.g. the choice of classifier and classifier parameters, need to be made beforehand without full knowledge about the data.
In this chapter the flow of events in the proposed system is visualized. The system identifies segments of a video that represent user specified events by using interactive learning without prior information about neither the video nor the events. First the online algorithm is described schematically in Paragraph 3.1, then an overview of the interactivity is given using the graphical user interface as a starting point. In Paragraph 3.3 the pre-processing steps are described. The goal of this chapter is to get a general overview of the possibilities and the building blocks of the system. All the blocks are explained in further detail in the following chapters.

3.1 Online algorithm

We divide the system in two parts: pre-processing steps and the online algorithm. The pre-processing steps have been taken out of the interactive process because of complexity issues. The goal of the pre-processing is to gather a set of good, general audio features. The online algorithm describes the steps that take place when a user is interacting with the system through the graphical user interface. Figure 3.1 represents a schematic view of the application already explained shortly using the user interface. The graphical user interface plays a selected video and shows a relevancy plot that the user can adjust and indirectly label samples this way so the classifier can train using the video frames and the relevancy values as labels for the video frames. There is no ‘best educated guess’ possible when the application is started because the user could select any event to be either relevant or not relevant. Therefore, when the user has not indicated any events -sets of frames- to be relevant or not relevant, the relevancy plot will only plot zeros.

From the raw video the features are calculated internally. There is a mapping between the feature vectors and the video frames. The relevancy plot can be changed by the user by pulling the relevancy plot up (important scene) or down (not relevant scene) at certain times. When the user has changed the relevance plot, the system can be (re-) trained. If the train button is pressed, the system will use the new labelled video frames as well as the frames from previous training iterations and use the relevancy plot as
labels to refine feature selection and the training of a classifier. Once the classifier has been trained, the relevancy plot will show the output of the classifier on the video frames that are playing, after some post-processing steps have been done in order to make the results more stable. This process of changing the relevancy plot and training the classifier can be done iteratively until the user is satisfied with the behaviour of the application.

![System overview diagram](image)

*Figure 3.1. System overview. From a raw video, features are extracted and a relevancy plot is shown. The user can give feedback to this relevancy plot and push a ‘train’ button. Then the features are used as data and the relevancy plot (including feedback) is used as labels for the relevancy classifier to be trained. After training, the relevancy plot is plotted based on this newly trained classifier and some post-processing steps.*

This online algorithm described in Figure 3.1 is an iterative process, so the user can give feedback on this new relevancy plot if necessary, train again etc., until satisfied with the behaviour of the classifier. More details about the features, the classifier and the post-processing will follow in Chapter 4, 5 and 6 respectively.

### 3.1 The user interface

A graphical user interface is constructed for playing a music DVD and this interface includes the feedback functionality for interactive learning. The user can play a music DVD and indicate the importance of some parts of this DVD in the interface. First this graphical user interface is inspected to get a feel for the information flow and the interactivity with the user. This user interface represents a simple, regular DVD player with some added functionality to handle the interaction via the plot in the bottom part of the interface and looks as follows:
In this user interface the user can open a movie using the top left ‘Open movie…’ button. This movie can be played and handled using the standard video buttons next to the video screen on the right. A progress bar shows which part of the movie has already been watched and shows the current frame number on mouse over. In the graph below a line appears along with the video. When the movie starts and no training has been done this graph will show just zero (‘undecided’) at every video frame. The user can drag this line to +1 (‘relevant’) or to -1 (‘not relevant’) for the video frames he or she thinks are relevant; this can be any event the user likes, for instance a specific instrument solo. Indirectly the training data is labelled in this way. When the user has labelled a few parts of the video the system can be trained by using the ‘Train’ button on the bottom right. When the ‘Play’ button is hit again after training, the system will now indicate in the relevancy graph whether the classifier thinks the current frame is relevant, not relevant or is indecisive. This is done for every frame displayed. When the user is not satisfied with the indicated relevancy, the graph can easily be adapted by dragging. This is the way of interactive communication between the user and the system: the system shows the user its knowledge in the relevancy graph and the user gives his/her feedback to the system by adapting this graph. The system can now be trained again, this time with more training information.

3.3 Pre-processing steps

Most of the pre-processing steps discussed in this section would in an ideal case be incorporated in the interactive process. Online feature extraction and integration are a possibility but the algorithm would not be able to perform all the operations in real-time any more. This is an issue of optimization and computational power and could be resolved in the future. For testing the concept it is important to have a real-time setting for the user to watch a video and interact with the system. The decision about which set
of general features is best can not be made without prior knowledge and the data and therefore had to be taken out of the loop. When a general set of features is available one could choose to use this and then the feature ranking would not be necessary. More thoughts about how to choose a general feature set without prior knowledge can be found in the discussion in Chapter 8.

Beforehand, 32 audio features are extracted from a video. From these features statistics are computed and those statistics are used as the actual features. This is called temporal feature integration. The statistical characteristics are: mean, standard deviation, minimum and maximum using a window of two seconds, overlapping one second. The features are discussed in Chapter 4. Per DVD there is a normalization step so every feature had zero mean and unit variance, to account for differences in volume and recording circumstances. Using forward feature selection all the statistical features are ranked according to their strength. This ranking is done for events like applause versus singing but also for different instruments. These different rankings are then combined - in a way that will be explained later- to get a sorted list of general audio features. A schematic view of the pre-processing steps is shown in Figure 3.3.

![Figure 3.3. Overview of the pre-processing steps: from the raw video, to the final feature set used for classification](image)

The offline feature selection discussed here results in a ranking of the features from which the top \( N \) features can be selected and used as a general starting set of features. In principle, when using interactive learning, offline feature selection is not necessary because when started with a large general set of features online feature selection can be performed in the interactive process. As a starting point we want to have a general set of features that is large enough to cover the entire content. We want to test if a smaller set of features to start with increases the efficiency of the algorithms. The results might be better because there is less correlation in the features and training and feature selection can be performed faster on a smaller training set.

More details about the basic features, the temporal integration of these features, the ranking and combining of the rankings can be found in Chapter 4.
Chapter 4.

Features

The input to our system is raw audio data: a signal that is the representation of the sound waves. This signal can be investigated in the temporal and the spectral domain and characteristics in these domains are captured in temporal and spectral features respectively. In this chapter we explain which features are used and how these features are calculated and selected. The accuracy of the classifiers depends on the underlying low-level representation of the audio data. The more discriminative the features, the better is the classification accuracy.

A list of current and promising new features have been composed based on the recent literature on audio classification on different topics. First this set of audio features is computed, which hold information on the audio signal extracted from a small to medium sized window. We tried our best to choose this combination of common and new general features to represent the music as effective as possible. This is feature set 1 and will be explained in Paragraph 4.1.

However, these short-time and medium-time features behave as a noisy time-series and in itself do not hold much structural information about the signal. Therefore a classifier will not yield the best results using these features directly. The time that a system has for the actual decision (decision time horizon) is often in the range of seconds instead of milliseconds and this larger time frame can be used for extracting temporal statistics from feature set 1. These temporal statistics often consist of mean and variance, as in [43][36][47]. This process of making new features on the larger time scale from the short-time features is called feature integration [24][43]. Experiments showed that this type of fusion (temporal feature integration) gave better results than using the short-time features for classification and performing a late fusion like majority voting between the sequences of outputs of the classifier on the short-time features [23].

Therefore temporal feature integration is performed on the first feature set by taking the mean, standard deviation, minimum and maximum of the first set of features along the temporal dimension over a so called texture window of 2 seconds, with one second overlap. This means that every window of 2 seconds will be used as a sample and is represented by a feature vector of length 128 consisting of the 4 statistical descriptors of every one of the 32 features from feature set 1. This new set of 128 statistical features
will be called feature set 2 and is discussed in Paragraph 4.2. This set will be ranked and used for classification in our system. This is visualized in Figure 4.1 and will be explained in detail in the next sections.

**Figure 4.1.** Two feature sets: one is extracted directly from the raw audio data and feature set 2 are statistical descriptors over the first feature set. The latter set is used as input to a classifier.

Firstly the short and medium time features will be explained in Paragraph 4.1, then the temporal integration is described and then the feature ranking and selection is discussed in Paragraph 4.3.

### 4.1 Feature set 1: short and medium time features

The first set of features consists of features calculated over a small and features calculated over a medium window. The short-time features are best calculated over a small window between 10 to 50 ms, depending on the complexity of the data. The medium-time features are calculated using a window of 600 ms, which is large enough to capture enough of the oscillations in the signal used for calculating these features. The medium scale has been selected to catch the perceptual information concerning timbre and modulation (instrumentation). The short time window is selected because the audio signal needs to be approximately stationary to compute these features and should represent the instant frequency (harmonics, pitch). We compute a set of set of 15 medium-time features per time window of 600 ms, with 400 ms overlap, and a set of 17 short-time features using a 30 ms time window, with 20 ms overlap.

These short and medium time features, listed in Table 4.1, are a combination of classical audio features and new features, based on crossing rates. The classical audio features consist of Energy, Pitch, Mel-Frequency Cepstral Coefficients (MFCCs) and Zero Crossing Rate (ZCR). The new features are higher order crossing rates and level crossing rates [26]. The features can be divided into temporal and spectral features. The
temporal features consist of the Energy Level, Zero Crossing Rate and the related Higher Order and Level Crossing Rates and the spectral features consist of Band Energy Ratio (BER), Pitch and the MFCCs.

<table>
<thead>
<tr>
<th>Feature:</th>
<th>Description:</th>
<th>Window length:</th>
<th>Domain:</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1:</td>
<td>Energy Level</td>
<td>short</td>
<td>temporal</td>
</tr>
<tr>
<td>F2:</td>
<td>Zero Crossing Rate</td>
<td>medium</td>
<td>temporal</td>
</tr>
<tr>
<td>F3-F9:</td>
<td>Higher Order Crossing Rates</td>
<td>medium</td>
<td>temporal</td>
</tr>
<tr>
<td>F10-F17:</td>
<td>Level Crossing Rates</td>
<td>medium</td>
<td>temporal</td>
</tr>
<tr>
<td>F18:</td>
<td>Pitch</td>
<td>short</td>
<td>spectral (temporal)</td>
</tr>
<tr>
<td>F19:</td>
<td>Band Energy Ratio</td>
<td>medium</td>
<td>spectral</td>
</tr>
<tr>
<td>F20-32:</td>
<td>MFCCs</td>
<td>short</td>
<td>spectral</td>
</tr>
</tbody>
</table>

Table 4.1: First set of 32 features with some characteristics. These will be further explained in the following paragraphs.

All these features will be explained in further detail in the next paragraphs. The ZCR is especially important because that is the basis for the higher order and band crossing rate features.

### 4.1.1 Energy Level

The energy level of a signal is also referred to as short time energy, root mean square, spectrum power, volume or loudness. It is the total spectral power of a frame and can be computed directly from the audio signal by:

$$STE = \frac{1}{N} \sum_{m=1}^{N} |x(m)|^2$$

(1)

Which is the short time energy of a window of size $N$. It is also possible to compute the energy level from the Discrete Fourier Transform (DFT) coefficients of a signal.

Among other things, this feature is used for distinguishing between speech and music signals.

### 4.1.2 Zero Crossing Rate

Zero Crossing Rate (ZCR) is a feature that is popular in audio feature because of its ability to distinguish between sound of different structures. ZCR is defined as the number of times the audio signal crosses the zero line. It is calculated as follows:

$$Z_0(n) = \frac{1}{2} \sum_{i} |\text{sgn}(x_i) - \text{sgn}(x_{i-1})|$$

where $\text{sgn}(x_n) = \begin{cases} 1 & \text{if } x_n \geq 0 \\ -1 & \text{if } x_n < 0 \end{cases}$

(2)
This is visualized in Figure 4.2, where an audio signal is plotted and a window is drawn of size 8 that holds 3 zero crossings.

![Image showing zero crossings in an audio signal](image)

Figure 4.2. Zero crossing rate visualized: There are 3 zero crossings in the window of size 8.

The zero crossing ratio is an indicator of the frequency at which the energy is concentrated in the signal spectrum. The strength of the ZCR is distinguishing signals that differ in this characteristic. It is often used for distinguishing voiced and unvoiced parts of speech. Voiced speech is produced by air streaming from the lungs through the vocal cords, setting them into an oscillating movement and form waves this way. These waves usually show a low zero-crossing ratio because of the periodicity. The unvoiced speech is produced by the constriction of the vocal tract narrow enough to cause turbulent airflow which results in noise and shows high zero-crossing ratio.

Besides separating voiced and unvoiced the ZCR can also separate a music solo from noisy applause: in solos the value of the ZCR will be a lot lower.

### 4.1.3 Higher order crossing rates

The spectral distribution of a signal can be extracted from the zero crossings of a signal and its derivatives. Higher order crossing rates are Zero Crossing Rates of the derivatives of the original signal. While the ZCR is an indication of the frequency of the signal, the HOCR are descriptive about the smaller variations in the signal.

As an example we have plotted a signal in Figure 4.3 and its first and second derivative in the next subplots. The zero crossing rate in the first subplot is now the ZCR, the crossing rate in the next two subplots are the higher order crossing rates because these are calculated over the first and second derivative of the signal. We can see the value of the ZCR in Figure 4.3 is 3, the first derivative looks at the smaller variations in the structure of the signal and counts around 35 zero crossings and the second derivative has around 90 zero crossings in this window.
Figure 4.3. An example of higher order crossing rates visualized. The ZCR and first two higher order crossing rate of a simple signal.

We computed 7 derivatives of the signal and computed the ZCR of these derivatives. The ZCR and the first two higher order crossing rates are shown in Figure 4.3. Higher Order Crossing rates can be used to distinguish between complex signals like different instruments, like the Level Crossing Rates [26].

### 4.1.4 Level crossing rates.

Level crossing rates or Band crossing rates combine information about frequency values and amplitudes. Not only is a sign change of the signal measured, but the transition from one amplitude region to another. The band is the amplitude region in the normalized signal between $-L$ and $+L$. A band crossing occurs when the signal value drops under $-L$ for the first time after the value was over $+L$. The following band crossing occurs only when the signal value exceeds the $+L$ value. This is visualized in Figure 4.4 below.

![Figure 4.4. Band crossing rate with $L = 2$. We see in this time frame of size 14 there are 5 band crossings.](image)
Level crossing rates combine knowledge about frequency and the volume or energy of the signal, which makes the information in this set of features is richer than the ZCR. Band crossing rates can be used to distinguish between complex signals like different instruments quite well, similar to the Higher Order Crossing Rates [26].

### 4.1.5 Pitch

Pitch represents the perceived fundamental frequency ($f_0$) of a sound. Pitch is loosely related to the log of the frequency, perceived pitch increasing about an octave with every doubling in frequency. Pitch can be estimated both in the time-domain and the frequency domain.

The most basic approach to the problem of $f_0$ estimation is to look at the original audio waveform over time and attempt to detect the $f_0$ from that waveform. There is a group of time-domain $f_0$ estimation methods which seek to discover how often the waveform fully repeats itself. The theory behind these methods is that if a waveform is periodic, then there are extractable repeating events that can be counted and the number of these events that occur in a second is inversely related to the frequency. The YIN $f_0$ estimator [6] is an example of a pitch estimator in time-domain.

There is much information in the frequency domain that can be related to the $f_0$ of the signal. Pitched signals tend to be composed of a series of harmonically related partials, which can be identified and used to extract the $f_0$. Many attempts have been made to extract and follow the $f_0$ of a signal in this manner. An example of a pitch estimator from the frequency domain is implemented in a speech analysis software package called Colea, developed by Philip Loizou [20].

### 4.1.6 Band Energy Ratio

(Sub) Band Energy Ratio (BER) is a feature that is based on the Energy Level feature as discussed in Paragraph 4.1.1. It can be obtained by dividing the frequency spectrum into sub-bands and computing the ratio between the Energy Level in the $j^{th}$ band and the total Energy Level of that window ($STE$). We only use the ratio between the low band $j$ energy compared to the total energy component.

\[
D = \frac{\sum_{m=1}^{H_j} |x(m)|^2}{\sum_{m=1}^{N} |x(m)|^2}
\]

where $L_j$ and $H_j$ are the lower and upper bound of sub-band $j$ respectively. BER is a feature that is used for general audio classification [19] because the spectral distribution of sounds from different sources differs quite a lot; for instance voices, background noise and music. It can be used for discriminating voiced and unvoiced audio samples because in a voiced signal the values of the signal energy tend to be in the low
frequency energy bands while other sources are more distributed over the frequencies [19][28].

### 4.1.7 MFCC features

The set of Mel-Frequency Cepstral Coefficients (MFCC) [11] are derived from a type of cepstral representation of the audio signal: a cepstrum.

First, the Fourier Transform of an audio signal is taken, after which the powers of the spectrum are mapped (using band-pass filters) onto a so-called Mel-scale. This mapping is performed because the human perception of hearing is very dependent on frequency of sound, so the range of frequency is not linear: for higher frequencies the ear is less sensitive then for the lower with the same sound intensity. In order to model human hearing in perceptual scales like the Mel scale are used. Then the logs of the powers at each of the Mel frequencies are taken. In the next step this Mel-scaled spectrum is transformed into MFCC using the Discrete Cosine Transform.

\[
C_n = \sqrt{2/K} \sum_{k=1}^{K} (\log S_k) \cos [n(k-0.5)\pi / K] 
\]

Where \(C_n\) is the \(n\)-th MFCC, \(K\) is the number of band-pass filters, \(S_k\) is the mel-scaled spectrum after passing the \(k\)-th band-pass filter and \(L\) is the order of the cepstrum. We compute \(L = 13\) MFCCs.

For the MFCC features it is important that the window is not too large because of the assumption of stationarity of the signal within that window. This stationarity can be approximated by using a small enough time window.

MFCCs form a set of features that is appreciated a lot in audio processing for their ability to differentiate between complex signals, but they are quite sensitive to noise. MFCCs are often used in speech recognition systems, such as the systems which can automatically recognize numbers spoken into a telephone [17]. They are also common in speaker recognition, the task of recognizing people from their voices [25][15].

### 4.2 Feature set 2: temporal integrated features

Feature integration in the temporal domain captures the structure in the audio signal better then single short-time features. The information from the time series of short-time features is integrated over a larger window into one new feature vector in order to capture the relevant temporal information in the window. The larger window is sometimes called ‘texture window’ because it holds information about the texture or structure of the audio signal.

The 32 features described in the previous paragraph do not hold enough information about the structure of the audio signal over longer time frames per single feature value,
but if the features are plotted in time-series, we can see a structure that stands for
different events [24][43]. This is shown in the Figure below where 2 minutes of Zero
Crossing Rate are plotted as a time-series.

![Image](image.png)

Figure 4.5. The Zero crossing rate of a part of a live concert DVD plotted as a time-
series. It can be seen that instrument solos have a low ZCR and a low standard
deviation, while applause has a higher minimum ZCR.

In Figure 4.5 we can see that there are time windows of different sizes in which there is
definitely an event recognizable. The values of the feature are individually not enough
to distinguish the events: for instance during solos the ZCR feature is very low, but low
values are also found elsewhere. The solos can be distinguished by a window of low
values, very low standard deviation of the values and also a low maximum. Applause on
the other hand has a high minimum in a large window.

Therefore temporal feature integration is performed on the first feature set by taking the
mean, standard deviation, minimum and maximum of the first set of features along the
temporal dimension. Every window of 2 seconds will be used as a sample and is
represented by a feature vector of length 128 consisting of the 4 statistical descriptors of
every one of the 32 features from feature set 1. This new set of 128 statistical features
will be called feature set 2. We let the texture windows overlap 50%. Because of our
scenario (we want to classify unseen frames based on the information in the frame and
the frames before because we cannot look into the future) the window used to calculate
a statistical feature consists of only previous frames. This means sometimes our
algorithm has a delay of maximal one second in finding out an event is occurring. The
online classifier will only have to be called every second now instead of every 30 ms.
This set of temporal integrated features is tested in Chapter 6 and utilized in the online
algorithm.

Because the features can have very different values and often classification algorithms
do not cope well with these different scales, a normalization step is needed to make sure
the scales are equal. Normalizing also takes into account differences in background
noise and recording volume per DVD. A standard normalization approach is to subtract
the mean of a feature from the feature values and then divide this by the standard
deviation, namely:
\[ X_i = \frac{(x_i - \mu_i)}{\sigma_i} \]  

(5)

where \( x_i \) is the \( i^{th} \) feature and the corresponding mean and standard deviation are calculated from the dataset. This normalization step makes sure that all the feature vectors get zero mean and unit variance.

4.3 Feature ranking & selection

In the previous paragraph we have derived a set of 128 statistical features; feature set 2. This feature set should be general enough for a range of audio events: users might be interested in selecting genres and styles, speech, applause, specific instruments, singing etcetera. This set can directly be used for classification.

Some issues arise if we want to use this complete set of features in our system:

1. We need low training and classification times. Maybe this feature set is too large for this
2. There might be quite some redundancy in the feature set
3. A user can specify any event he or she likes. Different events will be expressed in different subsets of the features.

We will first look at the possibilities for solving the first two problems by reducing the feature dimensions. This is an efficiency choice and will only effect the size of the offline constructed general feature set that was constructed in the previous paragraphs and will not be changed during the interactive process. We will check if the classification performance and the training speed improve significantly after this and if possible solutions fit in our scenario. This general feature set reduction will be discussed in Paragraph 4.3.1. To solve the third problem we want to perform online feature selection. This will be discussed in Paragraph 4.3.2.

4.3.1 General feature set reduction

In this paragraph we explore the possibilities to reduce the general feature dimensions that will be used because the current set might be too large and redundant. This is an efficiency choice and will only effect the size of the offline constructed general feature set that was constructed in the previous paragraphs and will not be changed during the interactive process. Besides the previously mentioned reasons there is another reason for doing this offline feature set reduction: in our iterative training scenario a user could select only a few samples (less than the number of features) to start training with, which will cause the system to over train severely if all 128 features are used. Still, this general feature set reduction will need to improve the efficiency significantly because the drawbacks of offline feature selection do occur in this case. If this is not the case and the feature set constructed in the previous chapter proves to be efficient enough, this offline feature set reduction is not necessary.
Offline feature reduction could provide us with the smallest dataset that is still general enough to identify a variety of events, but we do not know the size of this dataset, \( N \), and we have no complete prior knowledge about the events and the data. Because we do not know \( N \), we want to rank the features so the most important features are at the top of the ranking. Now we introduce the parameter \( N \) in our test sessions that holds the number of features that are used from the top of the ranking, so we can find out empirically what is optimal. The ranking should work for as many events as possible because the user is free to choose which events he finds relevant. This ranking could be done unsupervised, for instance using Principal Component Analysis, or supervised, using a forward feature selection algorithm. Both methods have some drawbacks that will be discussed.

One option is to only decorrelate the data, for instance by using Principal Component Analysis (PCA). This will minimize the redundancies in the data but it can not distinguish between good and bad features: for instance a feature that only adds noise and no information to the dataset will end up high in the ranking because it is very different from all the correlated, good features. This is because unsupervised methods only look at the possible amount of information that is present in a group of features but do not have any indication this information is valuable. On the other hand could we never use all possible labellings of the data to base our rankings on; this means it is just a simple heuristic to give a meaning to the features. Another disadvantage of using is a feature extraction method like PCA is the fact that it leaves us with a completely other-combined- feature set which is not comprehensible to users and developers while we think it is good to have a feeling for the features. We choose to use a feature selection method and not unsupervised feature extracting methods such as PCA.

Instead of PCA, feature selection is normally is performed to reduce feature dimensions and redundancy and hereby improving the efficiency of the feature set. To perform supervised feature selection a labelled training set is needed. Labels are linked to events; and events can vary and are unknown. Therefore this can only be approximated by creating several data sets that are labelled by a user for different events. The ranking is not optimal because not all of the possible events can be covered and the use of it should be seen as a heuristic. It also introduces prior knowledge about the events and has the disadvantages of offline learning. Multiple datasets have been labelled for different events. Per dataset forward feature selection will result in a ranking. We choose to use the labels of the following events to base our rankings on:

1. applause versus solo
2. applause versus singing
3. 2 different instruments (guitar and piano).

We use a sequential forward feature selection algorithm: a suboptimal feature selection algorithm that obtains a chain of subsets of features in straightforward manner. i.e. by adding the locally best feature in the set in terms of some criterion functions. In this report, the Mahalanobis distance is used as the criterion function for assessing the "goodness" of a feature subset. The Mahalanobis distance is a scale-free similarity measure for two sample sets based on correlation. Because the samples are highly correlated it is not possible to use the default criterion in PRTools, namely the 1-nearest neighbour error, for feature ranking: the nearest neighbour is (almost) always the next or
previous feature in the time-series and thus from the same class. The Mahalanobis criterion takes the whole distribution of the samples into account. This is measured by:

$$D^\text{Mahalanobis}_j = \sqrt{(x - \mu_j)^T \sum^{-1}_i (x - \mu_i)}$$  \hspace{1cm} (6)$$

Where $\sum^{-1}_i$ is the inverse of the covariance matrix of class $i$.

Now we have 3 rankings: one for every dataset that we labelled with the three different events. To make one general ranking we want to combine these rankings. From this combined ranking we can use the top $N$ features. In this paragraph we will try to find a suitable $N$.

### 4.3.1.1 Combining rankings

Combining the feature rankings of the different events can be done in 2 different ways:

1. For all 3 lists, give every feature a score that is its index in the ranking, and add up all the scores of the features and construct a new list, sorted ascending by score. The feature with the lowest cumulative score ‘wins’.
2. Take the best feature from all 3 lists and put these on the top 3 in the final list. Then take the seconds best features of all 3 lists and put these on the places 4 to 6 on the final ranking, etcetera. Then only keep the first occurrences in the list.

The first way of combining the rankings seems to be a fair and intuitive method, comparable to the scoring system of sports matches, like sailboat racing events. Multiple racing events take place in one season, and the most widely used way of calculating the season’s winner is the ‘Low Point scoring method’. It awards the winner his place in points - first gets 1, second gets 2, and so on, and adds up all points of all events. The overall winner is the team with the least points. An overall champion may often not have won a single event or a competitor may win three events but lose the overall title. In our scenario, we don’t know what events will be chosen by the user, but per set of events, we know the best features to distinguish the events; we only need to select them. But if we combine the rankings this way, these best features will be medium low on the final list, because they are not good at all to distinguish between other events. Features that are moderately well in distinguishing all classes will end up high in the list. Because we will perform an online feature selection step within the interactive process, we will need the best features from every list to be on top of the final list. That is why combining the features in the second manner is more rational in our scenario.

### 4.3.1.2 Results rankings

To verify our rationale about the combining of the rankings of the different events and to find a suitable size $N$ for the offline feature selection and a number of features $T$ to be selected online, we set up a test scenario. In this test scenario we select the top $N$ features from both off-line constructed rankings, where $N = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 128\}$. An online feature selection is performed to select the $T$ features from
this set that best represent the events targeted. This allows us to test which way of combining the rankings is better: the ‘minimum index’ or the ‘sum index’ combination.

With these selected features we do a ‘Leave One DVD Out’ cross validation on six DVDs. The classification procedure is as follows: on the training set of size $N$ the $T$ best features are selected using forward feature selection (Mahalanobis criterion), and with these a $k$-nearest neighbour classifier is trained and tested on the test set. For $T$ we have tried $T = \{2, 5, 10, 20, 30, 50, 128\}$, only combinations where $T \leq N$ could be tested. In the following table we present a small part of the results for the cross validation of the applause versus solo events:

<table>
<thead>
<tr>
<th>$N$ (# features offline selected)</th>
<th>$T$ (# features online selected)</th>
<th>Error rate method 1: sum index</th>
<th>Error rate method 2: minimum index</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>0.1403</td>
<td>0.1319</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>0.1782</td>
<td>0.1186</td>
</tr>
<tr>
<td>70</td>
<td>20</td>
<td>0.1716</td>
<td>0.1368</td>
</tr>
<tr>
<td>128</td>
<td>128</td>
<td>0.1558</td>
<td>0.1558</td>
</tr>
</tbody>
</table>

Table 4.2. Selection of the results (error rate) of cross validation of online feature selection of $T$ features from the top $N$ features from two rankings created by the two different combining methods. A $k$NN classifier is used. Method 1 combined by summing the indexes of the features and method 2 selected the features by taking the minimum index of the features for every event.

From the table it is clear that combining the rankings by method 2 gives better results: the error rate is smaller than the ranking achieved by applying method 1. This difference is also visualized in the Figures 4.7 and 4.8 a and b below. We plotted some results of cross validation over six DVD’s for both ranking methods for respectively three different datasets that represent the following events respectively: solo versus singing; singing versus applause and a dataset of different instruments.

In the left figures the number of offline selected features is static, which means we have taken $N$ features from the top of both rankings, and vary the number of features that are selected online, $T$, on the x axis for both types of rankings. In the right figures the number of online selected features is $T$ in every experiment and we varied the number of offline selected features, which means we have taken $N$ features from the top of both rankings, and from these features we select $T$ features every time in an online fashion.
Figure 4.7 a (left) and b (right). Part of the results for applause versus solo to compare the two types of rankings where a) the number of features online selected is varied and in b) the number of features that is selected offline is varied.

Figure 4.8 a (left) and b (right). Part of the results for applause versus singing to compare the two types of rankings where a) the number of features online selected is varied and in b) the number of features that is selected offline is varied.

The classifier used in these tests is a kNN classifier.

We can see that ranking 2, the minimum ranking, gives better results in most cases and on average. The instruments dataset was clearly harder to classify and the difference between the two rankings was not so clear. This is not a very surprising result as all the best features from all the events are in the top and will be selected during online feature selection. When \( N = 128 \) the results for both rankings are the same because then all features are used from the rankings so no offline feature selection takes place.

From the results of the piano versus guitar data it became clear that the number of train samples and the number of features were not sufficient to achieve a satisfying result. Especially the number of different DVDs to draw solo samples from should be more than three to be able to generalize; the number of features that were needed was a lot larger then the number of features needed to distinguish between applause and other events. This is shown in Figure 4.9, where all 128 features were used as a basis set for
online feature selection. This means there is no difference between the rankings anymore as all features are used and order is not important, so only one line is plotted. Even with the largest feature sets the results are not much better than random.

Two features for classification were sufficient for applause versus solo but clearly not enough for the instruments case. In the plots on the right of Figure 4.8 we varied $N$: the number of features selected offline. From these results it can be seen that an increasing $N$ does not produce a lower error rate for both rankings. This means it is possible to compute less features to start with, namely the top $N$ features, and this way reduce the time needed to compute all features in real-time while keeping enough information in this smaller set of features. Because time-complexity is not our main performance measure (as long as it stays within reasonable bounds for interactive use) and especially because there is prior knowledge necessary to construct this ranking and perform the offline feature selection we will refrain from offline feature selection and use all 128 features.

**4.3.1.3 Best features**

When the ranking is done using method 2 we can have a look at the top five features that are important per possible event. That set of features is chosen on the discriminative power of the features and the set is designed to have as little correlation as possible to account for redundancy by using forward feature selection. This was very important: because of the way we compute the features it is clear that there are a lot of dependencies and therefore redundant features. The top five of those sets of most important features is shown below.

![Figure 4.9. Error rate of a kNN classifier trained using cross validation over 4 DVDs on guitar versus piano data. Only one line is shown because the results of both methods are the same when $N = 128$.](image)
From these top lists we see that the families of Crossing Rate features are very good in separating different complex and less complex audio events from each other: These features help separate applause from more melodic parts of the audio and also help distinguish instruments from each other.

The best two features are plotted for applause versus singing for six DVDs in Figure 4.10 below to get a feel of the inter DVD behaviour of the events. This was the most separable case.

![Figure 4.10. Best two features are plotted for applause versus singing for 6 DVDs. There are two extremes visible and some overlap in between.](image)

### 4.3.2 Online feature selection

When a set of general features is available to start with, we still need to do online feature selection. From the set of $N$ features from the top of the ranking, $T < N$ features are selected using forward feature selection, performed online using a Mahalanobis criterion. This seconds feature selection step is necessary because the first set of features is very general and the system has to focus on the important features for a user specific event. If a user for example selects very high pitched audio parts as a target and no
Online feature selection is performed, the pitch is such a small part of the complete feature set that the other features will weigh too heavily in the nearest neighbour classifier to make this difference based only on pitch. Online feature selection will during interaction not only remove redundancy and reduce feature dimensions but also select those features that are relevant for the targeted events and focus the algorithm to those events this way. It then serves as a refinement of the feature set according to the events targeted. We do not know the optimum number of features $T$ to be used for this online feature selection. We expect that $T$ will not be the same for every event: some events will be more complex and therefore need more features to be able to distinguish them. As these events that are chosen by the user could vary a lot, it can be expected that online feature selection will improve the results. Online feature selection can benefit when a smaller, less redundant feature set is offered in terms of speed again, which is an advantage of combining both offline and online feature selection.

Selecting too little features will result in a lack of information to classify the more complex events while having too many features will not focus well enough on the specific selected events. The problem with setting this number is the lack of prior information, this is discussed in Chapter 8. When we use all 128 features the optimal set of online selected features is of size $T = 50$. This is found by averaging the results of Paragraph 4.3.1.2 over the three types of events; the optimal number of features will differ per targeted events but 50 features is a good compromise.

**4.3.3 Conclusion feature selection**

In conclusion we have made the following choices: the second ranking, constructed by taking the minimum index of the feature per event instead of the sum, was better. But: in our system no offline feature selection will be done as we need prior knowledge to construct the rankings and it is more important to us that the system is more general than a time-speed up. Online feature selection will be performed and $T = 50$ features will be selected.
Chapter 5.

Choice of classifier

In Chapter 3 the interactivity with the user is explained and the user feedback loop is formulated. The relevance feedback that the user provides is the information that will be used for constructing the new relevance curve. Relevance feedback has been widely studied for still images retrieval over the last years. Current techniques include feature weight adaptation and user query modification [48], though most of them use users labels to build a training set for classification. In this chapter we formulate the requirements of the classifier in our system. We then explore the possibilities of different classifiers in theory and test three types of classifiers in practice: a Support Vector classifier (SVC), a k-Nearest Neighbour classifier (kNNC) and a Logistic Linear classifier (LOGLC). The SVC classifiers were tested with multiple parameter settings and the kNN classifier has also been adapted to prior probabilities.

The choice of classifier is made based on the following requirements to the classifier:
1. Capability of learning efficiently from different sizes of training sets, especially small sets
2. Capability of handling unbalanced datasets. Most classifiers optimize overall performance. This can cause the classifier to be biased towards the larger class and the small set to be completely disregarded [21]
3. Capability of handling all different sorts of problems, from simple to complex class distributions; from separable to very overlapping because no assumptions are made about the data or events
4. Number of parameters to be set (using prior knowledge). The more complex a classifier is, the more information is needed beforehand to use this classifier.
5. Training speed. This is important for the interactivity. We do not need to pick the fastest classifier but there are limits to the training time.

We will first discuss the previously mentioned classifiers in terms of their strengths and weaknesses in relation to our problem in the next section. Afterwards the classifiers are tested in different scenarios.
5.1 Strengths and weaknesses of the classifiers in theory

Before subjecting the classifiers to practical tests using our music DVD data, we will first investigate the strengths and weaknesses of the classifiers in theory.

5.1.1 Support Vector classifier

Support Vector classifiers are well known modern classifiers and have a solid statistical background [44] as well as good practical results in various fields [4][7][29]. This is the reason for using this classifier in our system. A SVC can handle more complex problems than kNN and LOGLC, and might converge faster, but also has the disadvantages of a more complex classifier. A kernel and the associated parameters should be chosen. The radial basis function (‘rbf’) is a popular kernel for its good classification results in image retrieval and video content analysis [13]. If trained optimally, SVC produces very robust performing classifiers that outperform many other classifiers. The quality of the training though is dependent on the training data - as in other classifiers- but also on the parameters that need to be optimized, which is a difficult task. This problem is also recognized by Poulet: “There is a lot of papers published about the SVM algorithms and kernel methods, but very few of them address the parameters tuning to get the high quality results usually presented [. . . ] these results are difficult to reproduce because of the influence of the parameter settings.”[30]. Thus the extra task of choosing parameters could pose a problem, especially without prior knowledge about the videos. If on an unbalanced training set the regularization parameter C is not very large, a SVC simply learns to classify everything as negative because that makes the “margin” the largest, with zero cumulative error on the abundant negative examples. The only trade-off is the small amount of cumulative error on the few positive examples, which is only small [1]. A solution to this problem is formulated by Eitrich and Lang [9] who proposed a SVC in which the C parameter would be adjusted to the unbalanced dataset in such a way that two different C values are used, one for each class: C_i = C+ if the i
th training point is positive (label = +1), and C_i = C− otherwise (label = −1). In addition to correcting different sizes of the two classes, the (C+, C−) model can also account for different costs of false positive or false negatives. In our scenario a classifier is trained iteratively: when the result is not satisfactory, more training samples can be added. This poses another problem: the C parameter is sensitive to changes in the number of training samples, which is variable. This dependency is explained by the equation that is minimized within the SVC algorithm. The following figure visualizes a trained SVC.
The distance between the separating hyper planes in the SVC is $2/|w|$, so $|w|$ needs to be minimized in order to maximize the margin. In the case where the classes are not linearly separable, we need to introduce slack variables ($\xi_i$) to the equation to be minimized. This equation looks as follows:

$$\text{Min} \|w\|^2 + C \cdot \sum_i \xi_i \quad \text{such that} \quad c_i (w \cdot x_i - b) \geq 1 - \xi_i \quad \text{where} \quad 1 \leq i \leq n$$

(7)

Optimizing equation 7 means maximizing the margin while minimizing the error caused by the slack variables. This last error is weighted by $C$, which means that if the training sets are overlapping datasets and there are many training samples, this summation will be very large and in the end overshadow the maximization of the margin part of the equation. This could be solved partly by optimizing a $C$ parameter that is inversely proportional to the number of training samples. There is an implementation available that allows us to set the $C$ parameter differently for every sample. Using this possibility we can give a weight to samples and make sure our recall is boosted while precision is still reasonable, but we can also help the classifier focus more on the smaller class to enforce a good separation instead of choosing to classify every sample in one class. There is also a new implementation of the SVC in PRTools [8] that allows us to parameterize the maximum fraction of support vectors instead of the $C$ parameter, which indirectly could solve our problem partly. This implementation is used in our experiments and the number of support vectors is estimated by 1-NN estimation. Another version of the SVC is used for giving the two classes different $C$ parameters.

Support Vector classifiers are not so suitable for overlapping training sets, as another disadvantage of SVCs is that the assumption is made that the data is separable. If this is not the case a soft-margin classifier introduces slack variables because the samples in the overlapping area might be so many that the samples violating Karush–Kuhn–Tucker (KKT) conditions become abundant. Therefore the SVC has trouble finding a separating hyper plane in the case that the two classes overlap a lot. This causes the SVC to perform a lot of iterations which can cause a regular pc to have memory problems or to find a hyper plane that is not optimal at all.
An SVC can be trained iteratively, which could increase the speed in the following iterations because this is faster than retraining on the complete training set. Samples can be added one by one or in batches.

### 5.1.2 Nearest Neighbour classifier

A k-Nearest Neighbour (kNN) is a simple classifier that labels an unknown data sample by taking the majority vote of the k nearest neighbours. A kNN classifier can yield good results for our problem because of the behaviour of the features. Training is fast because all computation can be done during classification time: the distance of the test sample to all training samples is computed by computing the (Euclidean) distance between the vectors. In principle other distance measures could also be used. The label that the test sample will get is the majority vote of the k closest training samples. K can be set beforehand but can also be optimized by using cross-validation on the training samples: no need for parameter setting. Because the kNN classifier does not assume anything about the distribution of the data and it is sensitive to the local structure of the data it is a good option for our system that needs to work on any data and events.

Nearest Neighbour classifiers can be susceptible to noisy or irrelevant features and differences in feature scaling. Normalization and feature selection are therefore often needed. In unbalanced and overlapping training sets, the Nearest Neighbour algorithm has the tendency to classify a new vector to the class with the most training samples because these samples come up in the k nearest neighbours more often due to their large numbers. This problem of an unbalanced dataset could be solved by under- and/or over sampling respectively the largest or smallest class or by adapting the classifier, taking into account class priors. In the Support Vector classifier this could be done by adapting the C parameter for the different classes. In the kNN classifier this could be achieved by giving the samples from the small class larger weights in the majority vote: instead of choosing the label that occurs most frequently between the k nearest neighbours, the number of samples per class is first multiplied with the inverse of their prior before the maximum is chosen.

An example: a 2-class kNN classifier with k = 3 and an unbalanced training set with prior probability for class 1 (blue +) of $p_1 = 1/5$ and the prior probability for class 2 (red *) is $p_2 = 4/5$. To find the label of an unknown sample $x$ (green o) we find the k nearest neighbours $\{x_1, \ldots, x_k\}$ and count the number of samples per class in this set of nearest neighbours: $n_1 = 1$ and $n_2 = 2$. To adjust for the unbalance we multiply these numbers with the inverse of the class priors. Then the label of $x$ corresponds to the class with the maximum multiplication:

$$\text{Max} \left\{ n_1 \ast (1 - p_1), n_2 \ast (1 - p_2) \right\}$$

(8)
Figure 5.2. An example of a kNN classifier adapted to class priors. A normal 3-NN classifier would classify the unknown green sample into class 2 because 2 out of the 3 nearest neighbours belong to class 2. The weighted version we suggest here would classify the green point into class 1 because $2 \times 1/5 = 0.4$ is smaller than $1 \times (1-1/5) = 0.8$.

When we look at a larger unbalanced and overlapping problem we see that we can shift the nearest neighbour boundary such that the small positive samples are stressed, as visualized in Figure 5.3.

Figure 5.3. Decision boundary of normal kNNC (dashed line) compared to the adapted kNNC (full black line). $k = 3$.

We can conclude from Figure 5.3 that the error rate of the positive (blue) class will be lowered a lot because of this shift and the error rate of the negative (red) class will increase. Combined, the error rate will probably increase a little as the negative class is the larger class but the recall will improve a lot which in our case is more important.
5.1.3 Logistic Linear classifier

The Logistic Linear classifier (LOGLC) [2] is quite a simple, linear classifier without any user parameters for the user to specify. Computation of the LOGLC classifier for a dataset is done by maximizing the likelihood criterion using the logistic (sigmoid) function, which is the following function:

\[ y = \frac{1}{1 + e^{-x}} \]  

(9)

This function is plotted in Figure 5.4.

On the x-axis in Figure 5.4 we see one feature and on the y-axis the probability. We deal with a two-class problem with class 0 and class 1. The classifier maximizes the posterior probability of a feature vector given its class and the set of unknown parameters.

A LOGLC is added to the set of possible classifiers because of its capability to handle overlapping datasets very well. In our test events the data proved to be very overlapping which made it very hard for the Support Vector classifier to converge. A linear classifier is often used in situations where the speed of classification is an issue, since it is often the fastest classifier. Also, linear classifiers often work very well when the number of dimensions is large. Overfitting might occur when the classes are completely separable though, because the classifier will try to make the gradient in the sigmoid function infinitely large.

5.1.4 Classifier test setting

All of the above mentioned classifiers have their strengths and weaknesses. These will be tested on our music DVD data in the following sections. We could also reconsider the requirements and see if some problems can be solved outside the classifier. Especially the capability of handling an unbalanced dataset could be solved by sampling methods: the abundant class could be sub sampled or the small class could be super
sampled. When using a Support Vector classifier we can also adapt the C parameter per class so to punish errors in the classification of the small class more than the samples of the abundant class. The kNN classifier can also be adapted by incorporating prior knowledge as shown in Paragraph 5.1.2. This is important because if the dataset is very unbalanced it is logical for the classifier to classify all samples in the large class to achieve an error rate that is small: as small as the ratio small / large class. Logistic classifiers are not so sensitive to unbalanced datasets. The sub sampling solution might not work well if the training set is already small because sub sampling the negative training samples means a loss of information.

Among the classifiers we have tested are:
1. logistic classifier
2. normal kNN classifier, where k is optimized using cross-validation
3. adapted kNN classifier, where the unbalance of the training set is taken into account and k is optimized using cross-validation
4. SVC, nu algorithm, radial basis function kernel, parameter σ = 1
5. SVC, nu algorithm, polynomial kernel, parameter p = 1
6. SVC, polynomial kernel, parameter p = 1, C⁺ = 10⁴C⁻
7. SVC, polynomial kernel, parameter p = 1, C⁺ = 10⁶C⁻
8. SVC, polynomial kernel, parameter p = 2, C⁺ = 10⁴C⁻
9. SVC, polynomial kernel, parameter p = 2, C⁺ = 10⁶C⁻

We implemented all types of classifiers using PRTools to compare the results. Three different scenarios are tested:
1. Within one DVD:
   A user is watching a music DVD and selects applause segments and non-applause segments. The training set is enlarged iteratively: every iteration we add an applause segment and the non-applause segment in between. All seen samples are used. This is done for 11 applause segments in total and tested on a part at the end of the video containing different applause segments. This training approach suits our scenario best. Results are discussed in Paragraph 4.2.
2. Leave One DVD Out Cross-validation
   Per DVD there are two segments selected that contain applause and 2 segments that contain music solos. Of the 6 DVDs used we train on 5 and test on the last one and average this over the results of all 6 DVDs. This train approach shows how generic the results per DVD are. Results of this scenario are discussed in Paragraph 4.3.
3. Leave One DVD Out Cross-validation: Instruments
   Per DVD there are two segments selected that contain guitar solos and 2 segments that contain piano solos. Of the 4 DVDs used we train on 3 and test on the last one and average this over the results of all 6 DVDs. This train scenario is the hardest scenario because of the similarity of the two classes, the small size of the training set and because testing is done on a separate DVD. Results of this scenario are discussed in Paragraph 4.4.

The results have been measured in terms of the error rate and also precision (Eq. 10) and recall (Eq. 11) are calculated because of the imbalance of the number of positive and negative samples in the test sets. TP is the number of true positives, FP is the number of false positives and FN is the number of false negatives.
In our scenario, recall is more important than precision because we want to detect the – often rare – positive samples in the videos. Therefore the F5-measure (Eq. 12) is calculated that weights recall and precision in a single score. This is calculated as follows:

\[
F5\text{measure} = \frac{6 \times \text{precision} \times \text{recall}}{5 \times \text{precision} + \text{recall}}
\]  

(12)

Per scenario we trained the classifiers on the different increasing training sets using 50 features selected online from the complete set of 128 features. The main results per scenario are highlighted in the next three paragraphs.

5.2 Classifier results test scenario 1

The results of the first test scenario are presented here. Based on these results we will choose a classifier to implement in our system and, if necessary, the parameters for that classifier.

The scenario is as follows: A user is watching a music DVD and selects applause segments as his/her target and also some non-applause segments. The training set is enlarged iteratively: every iteration we add an applause segment and the non-applause segment in between. This is done for 11 applause segments in total and tested on a part at the end of the video containing 5 different applause segments. This training approach suits our application scenario best. The test set has been sub sampled to balance the positive and negative samples (261 samples each) to get interpretable results.
Table 5.1 shows the average results over all sizes of training sets for all nine classifiers on applause versus other data.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic linear</td>
<td>0.1371</td>
<td>0.9571</td>
<td>0.7736</td>
<td>0.7979</td>
</tr>
<tr>
<td>kNNC</td>
<td>0.1418</td>
<td>0.9302</td>
<td>0.8133</td>
<td>0.8279</td>
</tr>
<tr>
<td>Adapted kNNC</td>
<td>0.1231</td>
<td>0.9154</td>
<td>0.8711</td>
<td>0.8748</td>
</tr>
<tr>
<td>SVC, ‘rbf’, σ = 1</td>
<td>0.4997</td>
<td>NaN</td>
<td>0.0007</td>
<td>NaN</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1</td>
<td>0.1513</td>
<td>0.9295</td>
<td>0.8011</td>
<td>0.8154</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1, C+ = 10*C-</td>
<td>0.1357</td>
<td>0.9180</td>
<td>0.8450</td>
<td>0.8513</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1, C+ = 100*C-</td>
<td>0.1482</td>
<td>0.8938</td>
<td>0.8513</td>
<td>0.8516</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 2, C+ = 10*C-</td>
<td>0.1595</td>
<td>0.9041</td>
<td>0.8290</td>
<td>0.8310</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 2, C+ = 100*C-</td>
<td>0.1642</td>
<td>0.8999</td>
<td>0.8290</td>
<td>0.8296</td>
</tr>
</tbody>
</table>

We can see in Table 5.1 above that the adapted kNN classifier performed the best on average when we look at the error rate and the weighted combination of precision and recall, the F5 measure. Second best in performance was the SVC with a polynomial kernel of degree one, with a C parameter that was 10 times higher for samples of the smaller, positive class. The adaptation of the C parameter so the smaller class is punished more for misclassification effectuates in a higher recall compared to a non-adaptive C parameter. The logistic linear classifier scored best in precision. The SVC with a radial basis function kernel had too many problems with the unbalanced and overlapping training set and had the tendency to classify every sample into the majority class. This has been tested for multiple settings of the σ and C parameter, but seemed to be the case almost every time. Unfortunately we know that the results of the classifiers can be highly dependent on the data; we do not want to choose the best classifier for this applause versus other dataset, but the best performing classifier in general for live music DVD data.

In the following figures we have compared the results on applause versus other data of the different SVC classifiers visually, by plotting respectively the error rate, F5-measure, precision and recall on an increasing training set on the x-axis. The previous conclusions about the classifiers can also be drawn from these figures.
Different types of Support Vector classifiers were tested at first. The ‘rbf’ kernel was expected to give good results because of the many practical uses in literature with good results [33]. Unfortunately this was not at all the case (at least not for the default parameters that we chose), which was the reason for also trying a polynomial kernel. Almost all of the trained SVC with ‘rbf’ kernel classified all test samples into one, the largest, class, even when the C parameter was adjusted too punish the small class samples more for misclassification. No true or false positives were found, which resulted in an error rate of 0.5, zero recall and NaN precision and F5 measure. For the larger training sets this SVC yielded an “out of memory” error on a normal PC. As we
expected that one of the main reasons for the disappointing results of the ‘rbf’ SVC was the lack of optimization of the sigma parameter we have also tried to automatically optimize this parameter with the limited options we had within the training set. This was done by a PRTools function ‘RBSVC’, which computes a SVC classifier using a radial basis kernel with an optimised standard deviation by ‘REGOPTC’ that optimizes parameters by minimizing the weighted error rate of the classifiers estimated by using cross-validation. This classifier has not been taken into account in the tests in this chapter because of the training time: when the training set exceeded more than 500 samples the training time took longer than 10 minutes and was therefore unsuitable for the interactive scenario. Although not used for all tests some of the results of this optimized classifier for this first scenario were: on both small and larger training sets the error rate of this classifier was among the best scores of the other classifiers. The optimized sigma was around 10 to 20, which does explain the poor results of the other SVCs with ‘rbf’ kernels.

On this data the configuration that gave the best results was the SVC with a polynomial kernel of degree 1 and a C parameter that is 10 times as high for the small class then for the large class. This SVC is used in the other experiments to compare to the other classifiers. Learning curves of these classifiers are shown in the following Figures 5.6 a-d.

![Figure 5.6 a-b. Results of the classifiers on the first test scenario of an increasing training set within 1 DVD, tested on a separate later part of that DVD. a) error rate b) F5 measure](image)

F5 measure
Figure 5.6 c–d. Results of the different classifiers on the first test scenario of an increasing training set within 1 DVD, tested on a separate later part of that DVD. c) precision  d) recall

From the test results of the first scenario of classification of a part of a DVD based on previous parts of that same DVD, we can draw a few conclusions:

- The SVC with a ‘rbf’ kernel had trouble classifying these samples, even though this was one of the most separable test scenarios. This was caused by the not optimized sigma parameter; when optimized on the training set the results were comparable or even a little better than the other classifiers. Optimizing was intolerably time-consuming for larger training sets.
- We have found that the implementation of an SVC with a C vector that has 10 times larger values for the positive samples and a polynomial kernel of degree 1 gave results that were comparable to the kNN and logistic linear classifier and sometimes even better. The SVC seems too dependent on the context: when optimized the results are good, but there are a lot free variables to choose: type of kernel, kernel parameters, C parameter and then there is still dependence on the distribution of the training set.
- When the dataset became too large and overlapping the SVC results were not stable: not every run had a solution and there were memory issues, errors because the data points in the two classes were too similar or classified all data in one class.
- F-measure and F5-measure can be misleading measures to compare the classifiers with because of the quite high values when all samples are classified in the positive class, respectively 0.67 and 0.86 while the maximum score is 1.00. This can be even worse when the test set is unbalanced.
- A stable result of an error rate < 0.1 is feasible for this applause versus other dataset. This is calculated for all windows of 2 seconds of audio, without any post-processing. This means the final results could be better or worse, depending on the distribution of the true positives over the applauses.
- Not all applause scenes from the video are equally valuable in the training set: some applause parts give results that are worse when these parts are added to the training set, depending on the classifier.
- With these training sets in the order of thousands of samples and 128 features there is not much difference in the training time of the classifiers; the feature selection step usually took more time than training the classifier. When the training set was a lot larger we saw that the logistic classifier was fastest, a little slower was the kNN classifier because of the optimization of the k and the SVC classifier took a lot more training time.
- We conclude that the adapted nearest neighbour classifier is the best choice in this scenario. This adapted kNN classifier takes priors into account and especially boosted the recall at the cost of precision and overall error, which is an advantage in our scenario.

5.3 **Classifier results test scenario 2**

The choice of the adapted nearest neighbour classifier in the previous paragraph is supported by the results of the classifiers in the second test scenario presented here. To summarize the results of the second scenario we will use the applause versus solo test set, that is similar to the applause versus other test set used in the previous paragraph. The chosen samples are less overlapping than in the applause versus other test set because not all samples are taken into account but only pure applause and solo samples. The results are expected to be somewhat poorer because the test set is not a later part from the same DVD as the train samples, but from a completely independent test DVD. Training is performed on 5 DVDs and tested on one using ‘Leave One DVD Out’ cross validation. No offline feature selection is done; online feature selection selected 50 features from all 128.

Firstly, the cross-validated results of all classifiers in this scenario are stated in Table 5.2.

<table>
<thead>
<tr>
<th>Classifier:</th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic linear</td>
<td>0.2906</td>
<td>0.5879</td>
<td>0.5886</td>
<td>0.5378</td>
</tr>
<tr>
<td>kNNC</td>
<td>0.1621</td>
<td>0.8015</td>
<td>0.8122</td>
<td>0.7854</td>
</tr>
<tr>
<td>Adapted kNNC</td>
<td>0.1930</td>
<td>0.7192</td>
<td>0.8431</td>
<td>0.7972</td>
</tr>
<tr>
<td>SVC, ‘rbf’, σ = 1</td>
<td>0.3306</td>
<td>NaN</td>
<td>0</td>
<td>NaN</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1</td>
<td>0.1933</td>
<td>0.7597</td>
<td>0.7597</td>
<td>0.7385</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1, C+ = 10*C-</td>
<td>0.1933</td>
<td>0.7597</td>
<td>0.7597</td>
<td>0.7385</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1, C+ = 100*C-</td>
<td>0.2260</td>
<td>0.7101</td>
<td>0.6344</td>
<td>0.6339</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 2, C+ = 10*C-</td>
<td>0.2260</td>
<td>0.7101</td>
<td>0.6344</td>
<td>0.6339</td>
</tr>
</tbody>
</table>

*Table 5.2. Results for the nine classifiers in scenario 2: Leave One DVD Out cross validation.*
NaN values occur because in one of the test cases the classifier has resulted in a classification of all test samples in the negative class, which causes both precision and recall to be zero and a division by zero when the F and F5 measure are calculated. This is quite likely in the cases when only one or two DVDs are used for training and testing is done on a separate DVD: performance in terms of generalization is very poor then. On average the kNN has the lowest error rate, closely followed by the adapted kNNC and the SVCs with a linear kernel. When we look at the F5 measure we see that the adapted kNNC scores as the best classifier; even better than the normal kNNC because of the boosted recall that has a higher weight in the F5-measure. These cross-validation results were averages and we need to say the results have been very dependent on the distribution of the test DVD features. One DVD might be a little alike the training DVDs and the score became close to perfect; other DVDs were too different and the results got closer to random. The difference in genre seem to overrule the events we were looking for, which can be solved by a larger training set that is well distributed over the genres.

From the above test results of the second scenario of classification using Leave One DVD Out cross-validation we can draw a few conclusions:

- The kNN classifier performed best with an error rate of 0.16 and the adapted kNNC performed best in terms of the F5 measure: 0.80.
- In these first 2 scenarios where we tried to distinguish applause from other music DVD parts the results were better when fewer features were used: two features were sufficient. This may be caused by the simplicity of the problem and the correlation between the features.
- The SVC with ‘rbf’ with standard parameters gave the worst results again in this scenario.
- The SVC with ‘rbf’ kernel optimized for the training set performed quite well in the previous scenario but in this scenario, where generalization is tested, the results were a little worse than the classifiers that performed well in this case. This might be explained by the fact that there could be some overtraining on the training set because of the optimization.
- The SVCs with linear kernel performed quite well in this scenario, but still not as good as nearest neighbour classifiers. Performance of these classifiers could be better when the parameters are completely tuned to the current problem but this is not possible in our setting.
- The results of this between-DVD scenario are comparable to the results of the within DVD scenario (an error rate of 0.16 is feasible, which is 33% higher than the error rate of the first scenario of 0.12), which is promising in terms of generalization.

5.4 Classifier results test scenario 3

The choice of the adapted nearest neighbour classifier in the previous paragraph is also tested in an inter DVD scenario with events that are more complex to separate: guitar and piano solos. It is expected these events are hard to separate because the instruments
are both string instruments. Because the kNN classifier is one of the simplest classifiers available, we needed to test if the results were still better or at least comparable to the other classifiers in this case.

In a ‘Leave One DVD Out’ cross validation setting we trained the classifier on 3 DVDs and tested on another. On average the three training DVDs consisted of 240 samples. The average results are shown below in Table 5.3.

<table>
<thead>
<tr>
<th>Classifier:</th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic linear</td>
<td>0.5039</td>
<td>0.6263</td>
<td>0.6104</td>
<td>0.5830</td>
</tr>
<tr>
<td>kNNC</td>
<td>0.4780</td>
<td>0.6122</td>
<td>0.5017</td>
<td>0.5122</td>
</tr>
<tr>
<td>Adapted kNNC</td>
<td>0.4108</td>
<td>0.6009</td>
<td>0.7602</td>
<td>0.7250</td>
</tr>
<tr>
<td>SVC, ‘rbf’, σ = 1</td>
<td>0.6010</td>
<td>NaN</td>
<td>0.6667</td>
<td>NaN</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1</td>
<td>0.5190</td>
<td>0.6758</td>
<td>0.5225</td>
<td>0.5047</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1, C+ = 10*C-</td>
<td>0.5447</td>
<td>0.6550</td>
<td>0.5631</td>
<td>0.5257</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 1, C+ = 100*C-</td>
<td>0.5449</td>
<td>0.6752</td>
<td>0.3738</td>
<td>0.3804</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 2, C+ = 10*C-</td>
<td>0.4804</td>
<td>0.6489</td>
<td>0.4522</td>
<td>0.4690</td>
</tr>
<tr>
<td>SVC, ‘poly’, d = 2, C+ = 100*C-</td>
<td>0.4804</td>
<td>0.6489</td>
<td>0.4522</td>
<td>0.4690</td>
</tr>
</tbody>
</table>

Table 5.3. Average results over all tests for the nine classifiers in scenario 3, cross validation for guitar versus piano solos.

These results are really poor, this is not only caused by the previously mentioned similarity between guitar and piano, but also by the fact that three DVDs in the training set were probably not enough to generalize to the independent test DVD. This was quite DVD dependent as well: some classifiers achieved quite a reasonable error rate on some test DVDs but were completely wrong on another test DVD. It is nice to see that, even though the results are poor, the kNNC was the best classifier again and the adapted kNNC was able to improve the recall a bit. In fact, the only classifiers that could find some structure in the data and do better then random were both kNNC and the SVC with a polynomial kernel with a degree $d = 2$. The optimized SVC with ‘rbf’ kernel did not perform any better than the other classifiers with an average error rate of 0.52.

Concluding the search for classifiers, we have decided to use the adapted nearest neighbour classifier because there are no parameters, it was very quick during training and performed very well on all different sizes of training sets. The boosted recall is also a positive characteristic in our scenarios, while other classifiers have the tendency to have a low recall when the training set is very unbalanced (small number of positive samples). The results in both scenarios were better than all other classifiers tested. We will also implement the SVC with a linear kernel because those results showed promising results especially in the first scenario when a lot of training samples were available. We want to see how this popular classifier behaves in other scenarios because the results in these scenarios previously tested were disappointing.
Chapter 6.

Post-processing

In this chapter the post-processing steps that are taken are explained. The classifier that is trained will classify the test samples. There are several options for the classifier output that can be shown to the user in the relevancy graph. These options are drawn in Figure 6.1. This is the visualization of a part of a music DVD containing 4 relevant events, for instance applause.

![Possible classifier outputs](image)

**Figure 6.1.** Possible classifier outputs: crisp labels, soft labels or three classes that are based on the uncertainty. In the last option the most uncertain samples do not get a class label but are left 0.

In the scenario that we have in mind the user will watch a music DVD, labels some parts and then continue watching or searching, so the samples that follow in time will be labelled by the trained classifier. These labels are dependent - because the audio data is and the features are computed using overlapping windows- but classified independently.
per sample. Post processing can bring back the dependency and also add some prior knowledge about the events. We have the choice to use ‘crisp’ labels for classification or ‘soft’ labels. Crisp labels are the conversion from the classifier output to the class labels. Soft labels are the normalized classifier outputs such that the sum of the outputs for one sample equals one so these values can be used as posterior probabilities. These different types of classification ask for different types of post-processing: crisp labels can be used for a straightforward way of post-processing, while soft labels give us the opportunity for a more sophisticated post-processing procedure that takes into account the uncertainty of the classifier. Both types of post-processing are implemented and will be discussed in the following paragraphs.

We have to note that it is possible to boost the results of the separate classification steps quite a lot with the right post-processing steps, but that if we stick to our scenario the information needed to optimize the results is not available. This is the case for smoothing and majority voting for instance where the optimal results would be achieved by using a normal moving window but in our case we can only use samples from the past that we have already seen which causes a delay in detecting of the events. Another issue that could boost our results is prior knowledge about the events: an example of this prior knowledge is that an applause is always longer than five seconds and never has a very short period of instrument solo in between. We can not use this prior knowledge because it should work with all events people choose, but we can make sure the classifier outputs are stabilized such that there are no quick changes from one event to the other. So the lack of an independent test set and prior knowledge about the data and the events and the fact that we can only look back in the data results in quite some restrictions in the post-processing options. Results of both crisp and soft classification are analyzed in Chapter 7.

6.1 Crisp classification post-processing

The classifier that is trained will classify the following samples into crisp labels: 2 classes: 1 (negatives) and 2 (target, positives). These labels are dependent but classified independently. Post processing can bring back the dependency and also add some prior knowledge about the events. We do introduce a dependency to the data and events. When crisp labels are used the post-processing consists of two steps:

1. Smoothing
2. Rounding

Each of these steps has a parameter. These parameters are chosen manually and not optimized because optimization is done with respect to the data and the events. If that would be available beforehand, these parameters could be optimized by a simple grid search. First the sample labels are smoothed by using a moving average filter of size 8 using only labels of previous samples. This smoothing step introduces dependency in the classification of the labels and therefore makes sure the labels will not jump up and down. These 8 feature vectors correspond to 8 seconds of music video. The averaged values need to be rounded to get back to class labels again. Because the introduction of a dependency based on previous samples also causes a delay in the detection of events
and because recall is important in our scenario, rounding is parameterized. When the labels are 1 (negatives) and 2 (target, positives) after smoothing all samples > 0.4 are rounded up instead of rounding up from 0.5. This is another variable that is dependent on the distribution of the events. In the surveillance scenario it is clear that the dataset is unbalanced and that recall is more important than precision. In the current music DVD scenario this is variable and without knowledge about the data or events impossible to optimize.

We will present some results of the post-processing based on the nearest neighbour classifiers. The adaptation of the nearest neighbour classifier that has been discussed before can also be seen as a sort of (soft) post-processing. The training set consisted of the first two sets of applauses of a music DVD (544 samples) and the test set consisted of the last samples of the same music DVD (2600).

<table>
<thead>
<tr>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNNC</td>
<td>0.02995</td>
<td>0.9480</td>
<td>0.7307</td>
</tr>
<tr>
<td>Adapted kNNC</td>
<td>0.02884</td>
<td>0.8679</td>
<td>0.8281</td>
</tr>
<tr>
<td>Post processed adapted kNNC</td>
<td>0.02745</td>
<td>0.8765</td>
<td>0.8338</td>
</tr>
</tbody>
</table>

*Table 6.1. Results of a kNNC, adapted kNNC and post processed adapted kNNC within one DVD.*

We can see in the Table above that the adapted kNNC boosts the recall and has a lower precision than the normal kNNC. In this example even the error rate is a little lower, but normally the error rate of the adapted kNNC is a little higher; a trade-off that boosts the recall at the expense of precision and error rate. The reason that the error rate is lower now is because the test set has been subsampled to achieve a balance between the positive and negative samples. Normally the gain would be in the small percentage of positives while the loss is in the large group of negative samples, which weights more in the error rate. The F5 measure of the adapted kNNC is higher then the F5 measure of the normal kNNC. In this case post-processing does improve the results a little, this is not always directly translated into better error rate or F5-measure but stabilizes the results.

This example of crisp label post-processing and the differences per setting are visualized in Figure 6.2. First the labels of a regular kNN classifier are plotted on top of the true labels, below the adapted kNNC labels are plotted and the bottom plot shows the results of a post processed adapted kNNC.
We can see in Figure 6.2 that most of the time the classifier finds the applauses (8 out of 9). There are a few false positives in the beginning of the test set, especially when the adapted kNNC is used. It can be seen that the adapted kNNC classifies more samples as ‘applause’ than the normal kNNC. Besides boosting the recall this adaptation introduces some false positives too. Part of the false positives is removed in the post-processing steps, as can be seen in the bottom plot.

### 6.2 Soft classification post-processing

There is a possibility to get an indication of the posterior probability of the classes in PRTools for every classifier. Using these probabilities gives us more information about the uncertainty of the classifier on the label of a sample. These probabilities can be used to ‘remove’ uncertain samples from the post-processing stages. Soft classification post-processing in already incorporated in some sense in the adapted kNN classifier. The adaptation that takes the class priors into account actually can be seen as a change of the rounding parameter of the soft labels. Therefore soft classification processing will be implemented for the Support Vector classifier and the standard kNN classifier. When
the steps are exactly the same as the crisp label post-processing procedure discussed in Paragraph 5.1, namely smoothing and –parameterized- rounding, only applied to the soft labels, the uncertainty of the classifier is taken into account per sample.

Another possibility that is introduced while using the soft labels is the addition of a ‘class’: samples that are certain enough of their class are classified into the ‘relevant (1)’ or ‘not relevant (-1)’ class, but if the classifier is too uncertain about a sample it can also be classified into the ‘uncertain’ class. It is evident that this should not happen too often in a reliable classifier but it is good to admit the weaknesses of the classifier instead of more or less guessing a label because the user only sees the labels and not uncertainties and therefore judges all labels as equally reliable. This procedure is different from the crisp label post-processing in the rounding of the smoothed samples: after smoothing the uncertain samples with probability values between 0.3 and 0.5 to be of the positive class are discarded because they are too uncertain, values over 0.5 are rounded up (positive class) and values below 0.3 are rounded down (negative class).

An example of both soft label post-processing options is shown in Figure 6.2 below. The training set consisted again of the first two sets of applauses of a music DVD (544 samples) and the test set consisted of the last samples of the same music DVD.

<table>
<thead>
<tr>
<th></th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>kNNC</td>
<td>0.02995</td>
<td>0.9480</td>
<td>0.7307</td>
<td>0.7597</td>
</tr>
<tr>
<td>Post processed kNNC</td>
<td>0.02413</td>
<td>0.8619</td>
<td>0.8940</td>
<td>0.8885</td>
</tr>
<tr>
<td>Post processed kNNC with uncertainty class</td>
<td>0.01312</td>
<td>0.9145</td>
<td>0.9360</td>
<td>0.9324</td>
</tr>
</tbody>
</table>

Table 6.2 Post processing soft labels results for a normal kNNC, soft labels post processed kNNC and a soft labels post processed kNNC that labels samples with ‘uncertain’ instead of one of the classes if the classifier is too uncertain.

For this data the soft label post-processing gave good results; better then the crisp label post-processing that got the best results on the post processed adapted kNNC: an error rate of 0.02745 and F5 measure of 0.8406. This is because smoothing takes the uncertainty of the classifier into account and is therefore more accurate. The ‘Post processed kNNC with uncertainty class’ gave the best results but this is not completely fair to compare as around 5% of the data was given the label ‘uncertain’ and these hard to classify samples are not taken into account in the computation of the results of this classifier. It is still an advantage though for a classifier to indicate to a user when it is just too uncertain to give a label to a sample. This classification of the three just evaluated classifiers is visualized again in Figure 6.3 below.
Figure 6.3 Post processing soft labels results for a normal kNN classifier, soft labels post processed kNN classifier and a soft labels post processed kNN classifier that labels samples with ‘uncertain’ instead of one of the classes if the classifier is too uncertain.

We can see in Figure 6.3 that most of the times the classifiers find the applauses (8 out of 9). There are less false positives than when the crisp post-processing was done, but it can be seen from the bottom figure that these have changed from a false positive to a ‘uncertain’ sample. In this example of soft label post-processing we can see the uncertain samples, because we gave them the label 0.5. Often these uncertain samples are found around the events, the on- and offset of an events are often unclear to classify or even overlapping in the 2 second windows. The parameters involved for choosing how uncertain a classifier needs to be to classify a sample as uncertain (or relevant or non relevant), are influencing is a trade-off between recall and precision and the number of (allowed) ‘uncertain’ samples.

We choose to give the user feedback in three classes when soft labels are used: ‘relevant’, ‘irrelevant’ and ‘uncertain’ to keep the interface and the choices for the user simple and clear. Direct plotting of the uncertainty of the classifier might be confusing for the user as the feedback can directly be used as labels for the following training iteration.
Chapter 7.

Results

In this chapter we have tested the most promising techniques from the previous chapters extensively. As no offline feature selection has been done, all 128 features are used in these experiments, refined by an online feature selection that selects 50 features from this set. The online feature selection is forward feature selection and the optimization is based on the maximum value of the sum of estimated Mahalanobis distances. The following classifiers are tested:

1. Support Vector classifier with a polynomial kernel of degree 1 and $C' = 10^4 C'$, crisp labels post processed
2. kNNC classifier adapted to prior probabilities in the training set, crisp labels post processed. $K$ is optimized by ‘Leave One Out’ cross validation
3. kNNC classifier, soft label classification and post-processing with an ‘uncertain’ class. $K$ is optimized by ‘Leave One Out’ cross validation
4. A standard kNNC classifier is also tested for comparison.

The classifiers are tested automatically on a few events, that have been suggested by possible users in a brainstorm session. These automatic tests performed in Paragraph 7.2 and 7.3 have focussed on the classifier performance and the interaction with the user is simulated. The classifiers are tested in a within DVD scenario, discussed in Paragraph 7.2, and a ‘Leave One DVD Out’ cross validation scenario discussed in Paragraph 7.3. In the last test setting, discussed in Paragraph 7.4 we have tried to benefit the most from the interaction with the user. This is done by sampling the training set in such a way that new samples are stressed during training because these are indications of the user to the classifier where classification still goes wrong and in this way these new samples are more valuable. A new parameter is introduced that holds the number of iterations that a sample has already been in the training set and is used for stressing the new samples. This is a within DVD scenario, but the set up is different from the set up in Paragraph 7.2. The difference is that this last scenario takes the errors of the classifier and the reaction of the user into account, while the first scenario only simulates the increasing training set.
7.1 Measurements

In a within DVD scenario, two ways of measuring the results are adopted: per sample and per applause. Together these measurements give a good idea of how good the classifier plus post-processing results were. Only looking at per sample results does not give any information about events that were completely missed, while only looking at events makes it easy to score high by enlarging the parts classified as positive (applauses for instance). It is important to look at the combination of both measures to get a good impression of the results for the methods. The way true positives, false positives and false negatives are measured when we look at the events as complete entities is visualized below.

A true positive is when the classifier found the event in some time span and this time span overlaps with the true time span of that event, as shown in Figure 7.1 a. A false positive occurs when there is no overlap between a detected event and a true event; this is visualized in Figure 7.1 b.

![Figure 7.1 a and b. Figure 7.1 a (left) shows how to measure the true positive events and Figure 7.1 b (right) shows how to measure the number of false positive events.](image)

A false negative occurs when a true event is not detected at all, this is visualized in the figure below.

![Figure 7.2. A visualization of the false negatives when results are measured per event.](image)
The error rate, precision, recall and F5 measure will be calculated using both the previously defined measures and the standard per sample measures. In a within DVD scenario, it is sensible to analyze the combination of both measures: results per sample and per event, to get a fair impression of the results for the methods.

Because all tests consist of a series of smaller tests, ROC curves of the overall results can not be plotted. In the within DVD scenario the training set is increased to simulate every iteration. In the inter DVD scenario the train and test set are differed by cross validation.

### 7.2 Scenario 1: within DVD

In this within DVD scenario the classifiers are tested on the first part of a DVD and tested on the last part of the DVD. The relevant DVD sections in this paragraph are the applause sections; all other events have been labelled irrelevant. The training part is extended with one relevant event every iteration. This is done over three complete DVDs and the results are averaged over these DVDs, which are: Andrew Bird (total 4605 samples), Spinvis (total 6536 samples) and Blues Brother Castro (total 2092 samples). The classifiers are trained on the first 1,2,3,4 and 5 applauses and tested on the final 40% of the samples of the corresponding DVD.

This scenario is visualized in Figure 7.3.

![Figure 7.3. Visualization of the simulated test scenario. A time line is shown with a few relevant (+1) events. In the first iteration the first non applause plus the first applause section is labelled by the user. In the second and third iteration the train set is extended until the next applause. After every iteration the classifier is trained. Testing is performed on the last section of the DVD.](image)

Table 7.1 shows the results of this scenario, averaged over all test DVDs and training set sizes.
Table 7.1. Results of the four chosen classifiers on applause versus other, averaged over the three test DVDs and the 5 training set sizes.

<table>
<thead>
<tr>
<th></th>
<th>Error rate per event</th>
<th>Error rate per sample</th>
<th>Precision per event</th>
<th>Precision per sample</th>
<th>Recall per event</th>
<th>Recall per sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal kNNC</td>
<td>0.2189</td>
<td>0.0386</td>
<td>0.5019</td>
<td>0.6905</td>
<td>0.8156</td>
<td>0.6751</td>
</tr>
<tr>
<td>SVC ‘p’,1, C+= 10*C- post proc.</td>
<td>0.2256</td>
<td>0.1208</td>
<td>0.5518</td>
<td>0.4779</td>
<td>0.6295</td>
<td>0.6295</td>
</tr>
<tr>
<td>Adapted kNNC, post proc.</td>
<td>0.1912</td>
<td>0.0622</td>
<td>0.5302</td>
<td>0.5378</td>
<td>0.7781</td>
<td>0.7277</td>
</tr>
<tr>
<td>kNNC, pp. incl 'uncertain'</td>
<td>0.1625</td>
<td>0.0361</td>
<td>0.6451</td>
<td>0.6331</td>
<td>0.7684</td>
<td>0.8086</td>
</tr>
</tbody>
</table>

We can see in Table 7.1 that the Support Vector classifier results were worse than a normal nearest neighbour classifier. The post processed nearest neighbour classifier gave the best overall results. The error rate of this best classifier is plotted for the 5 different training set sizes in Figure 7.4 to inspect the learning behaviour.

Figure 7.4. Learning curve of the kNN classifier on applause versus other, post processed with an uncertain class. The error rate per sample and per event are plotted against the number of applauses in the training set, again averaged over the three test DVDs.

The error rate per event (the red line) is higher than the error rate per sample. The reason this is consequently higher can be found in the fact that the ‘per event’ set is not unbalanced and therefore all rightly classified negative samples do not weigh as strong as in the ‘per sample’ calculations. It indicates that the classifier still misses some of the applauses completely. Usually these are the short applauses; sometimes the band is playing though a large part of the applause. These applauses seem to be too hard to classify no matter how large the training set because the overlap with the other class is just too large.
7.3 Scenario 2: inter DVD

In this second scenario the generalizability of the system is tested by training the system on samples from several DVDs and testing the resulting classifiers on samples from another DVD. This is done by ‘Leave One DVD Out’ cross validation on 6 DVDs, in which parts of the DVDs have been labelled as applause, instrument solo, singing, guitar and piano. The classifiers are subjected to training and testing of the following events:

1. applause versus instrument solo
2. applause versus singing
3. guitar versus piano

The results of these tests will follow in the next sections, in terms of error rate, precision, recall and F5 measure, calculated per sample.

We start by presenting the results of applause versus solo. The training set consists of around 300 samples and is reasonably balanced. Results in the following table are averaged over all test DVDs in the cross validation.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5 measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal kNNC</td>
<td>0.1377</td>
<td>0.8220</td>
<td>0.8122</td>
<td>0.7941</td>
</tr>
<tr>
<td>SVC ‘p’,I, C+=10*C- post proc.</td>
<td>0.1750</td>
<td>0.7303</td>
<td>0.9375</td>
<td>0.8869</td>
</tr>
<tr>
<td>Adapted kNNC, post proc.</td>
<td>0.1675</td>
<td>0.7490</td>
<td>0.9271</td>
<td>0.8656</td>
</tr>
<tr>
<td>kNNC, pp. incl ‘uncertain’</td>
<td>0.1656</td>
<td>0.7573</td>
<td>0.9405</td>
<td>0.8802</td>
</tr>
</tbody>
</table>

Table 7.2. Cross validated results of the four chosen classifiers on applause versus solo.

We can see that the normal kNNC produced the lowest error rates, but the post processed versions achieved a higher recall and therefore a better F5 measure; a property that is important in our scenario. In this test setting the recall and F5 measure are a little less informative because the test set has been produced by taking some parts from the DVDs in such a way that it is balanced. Boosting the recall means classifying more samples as positive, this is proportional to the number of positives in the set. When the test set is artificially balanced, the gain in recall translated in a significantly higher error rate. The post processed SVC had the largest error rate and the largest tendency to classify samples as positive; this results in a high recall and a high F5 measure. This is not so strange as the SVC has been adapted to unbalanced datasets, not only in post-processing but in the parameters of the classifier as well.

The results of applause versus singing are a little better than applause versus solo and are mentioned in the table below.
<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5 measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal kNNC</td>
<td>0.1106</td>
<td>0.9505</td>
<td>0.7628</td>
<td>0.7766</td>
</tr>
<tr>
<td>SVC `p',1, C+= 10*C- post proc.</td>
<td>0.1152</td>
<td>0.8231</td>
<td>0.9033</td>
<td>0.8786</td>
</tr>
<tr>
<td>Adapted kNNC, post proc.</td>
<td>0.1107</td>
<td>0.8469</td>
<td>0.8690</td>
<td>0.8551</td>
</tr>
<tr>
<td>kNNC, pp. incl 'uncertain'</td>
<td>0.0923</td>
<td>0.8488</td>
<td>0.9333</td>
<td>0.9085</td>
</tr>
</tbody>
</table>

*Table 7.3. Cross validated results of the four chosen classifiers on applause versus singing.*

As mentioned before these results were better than on the applause versus solo dataset, getting error rates below 0.1 and an F5-measure > 0.9. Here the kNNC that used the ‘uncertain’ class clearly got the highest scores. The SVC scored well on the F5 measure because it was the most adapted to boost the recall, which is exactly what it did but the results are based on the balanced test set thus the effect of the post-processing is very large. The adapted kNNC has performed well on this set; the recall/precision ratio is changed to our likings compared to the original kNNC without compromising significantly on the error rate.

We have noticed that genre differences had some influence on these inter DVD results: some DVDs had a distribution that was too different from the training set and has much higher error rates than other DVDs. To indicate the way classifiers handled this difference between DVDs we have also plotted the error bars of the different classifier so the error rate can be linked to the standard deviation. In Figure 7.5 it is clear that all of the classifiers did not have a problem classifying the applause and singing samples from the P. Wolff DVD while the Spinvis DVD gave a lot more trouble. It is also shown that genre and probably instrument choice had a large influence on the applause versus solo results. The train set clearly did not generalize enough to classify the samples from Ane Brun. All of the errors made we from the solo class.

![Error bar chart](image_url)

*Figure 7.5. Error bar of the inter DVD tests per music DVD to show that the results depended a lot on the distribution of the test DVD.*
To see which classifiers gave the most stable results we also plotted the error bars per classifier in Figure 7.6.

![Error bar per classifier](image)

*Figure 7.6. Error bar per classifier of applause versus singing in the inter DVD scenario.*

The error bar per classifier shows that although there is not much difference on average—especially between the normal kNN classifier and the postprocessed SVC—the standard deviation of the normal kNN classifier is a lot larger.

The last results in this scenario are the results of the guitar versus piano dataset. Less training samples from less DVDs were available in this scenario, around 200 samples from in total different 3 DVDs.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F5 measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal kNN</td>
<td>0.5308</td>
<td>0.5874</td>
<td>0.4085</td>
<td>0.4206</td>
</tr>
<tr>
<td>SVC ’p’,1, C+=10^4C- post proc.</td>
<td>0.4225</td>
<td>0.6860</td>
<td>0.4323</td>
<td>0.4555</td>
</tr>
<tr>
<td>Adapted kNNC, post proc.</td>
<td>0.4676</td>
<td>0.6147</td>
<td>0.4911</td>
<td>0.5056</td>
</tr>
<tr>
<td>kNNC, pp. incl ‘uncertain’</td>
<td>0.4551</td>
<td>0.8618</td>
<td>0.4997</td>
<td>0.5150</td>
</tr>
</tbody>
</table>

*Table 7.4. Cross validated results of the four chosen classifiers on applause versus singing.*

Even though the results are really poor, it is interesting to see that the SVC got the lowest error rate in this more complex case; post-processing got the error rate down a bit and the post processed kNN classifiers performed best when looked at the F5 measure.
7.4 Scenario 3: within DVD, stressing new samples

This scenario is added to see if the system can profit more from the interaction with the user and what happens if the testing is more realistic in terms of human users. In a realistic scenario, a user will not label all samples, but just some parts that are definitely relevant and some that are not. Then the classifier will be trained and the video will resume playing from where it was paused before training. Then the user will keep watching until the classifier classifies something wrong, which will be immediately corrected by the user. A good example is when only a few applause samples are selected as being relevant, the classifier will also classify the first drums solo as applause as well because of the similar acoustics, which is what happened in the Andrew Bird video that is user for all tests in this scenario.

The test scenario is the following: A user watches a DVD and selects the first applause as relevant and a few surrounding samples as being irrelevant. The system trains a classifier using these samples labelled by the user in its training set. The kNN classifier adapted to the prior probabilities is used in the following test, without post-processing so we can see the pure results of the classifier. Then the system shows the user the relevancy plot based on the labels given by the classifier and the user will adapt the relevancy plot when something is labelled wrongly. Then the classifier is trained again using this new information, this is one training iteration. The following plot shows the results of 6 training iterations, in which every iteration 15 consecutive labels are added/changed by the user. These are added in the DVD section in which the classifier makes the first error in the relevance curve, for instance a false positive. The scenario is visualized in the following figure:

![Figure 7.5. Visualization of the third scenario. A time line is shown with a few relevant (+1) events is shown. In the first iteration the first non applause plus the first applause section is labelled by the user. In the second and third iteration the user adapted a few samples where the classifier went wrong. User feedback is indicated by the green line. All other previous samples have been labelled indirectly by the previous classifier. After every iteration the classifier is trained. Testing is performed on the last section of the DVD. ](image-url)

We can choose to use all samples that have been seen in the training set; this would be all samples that the classifier classified correctly and all samples that have been corrected by the user together. This is what has been done in the previous tests. But there is some addition information in this training set that could be useful: it is known
now which samples were hard to classify - the user feedback- and which weren’t - the sampled classified by the previous classifier. Another source of information is the time that the samples have been in the training set already, this information is not so useful in the music DVD case but can be valuable when the system is developed for smart surveillance systems. Older samples can be ‘forgotten’ after a while to help the system adapt to new circumstances and treats. This is not tested here but would be valuable when the dataset becomes very large.

For testing the impact of using this information several sampling methods have been tested on the applause versus other scenario. The following sampling methods have been subjected to testing:

1. All samples so far are used. The 15 samples that the user labelled manually are super sampled twice to stress these samples (black line)
2. Every iteration 15 samples are added randomly, chosen from the samples seen so far (red line)
3. Every iteration simply the following 15 samples are added, without regarding the user feedback (green line)
4. All samples so far are used, regardless whether these were classified by the previous classifier or adapted by the user (dark blue line)
5. Only the 15 samples per iteration that the user labelled manually were added to the training set.

To clarify these sampling methods these are visualized in Figure 7.6.

![Figure 7.6. The different sampling methods visualized. In the first iteration all methods are the same because the user initiates the process by selecting a few samples. The user labels 15 extra samples each iteration. These samples are chosen and combined into training sets in different ways.](image)

The results of these tests are shown in Figure 7.7 below. Super sampling is artificially creating new data from a specific class. In our case we super sample the user feedback samples and is implemented by simple data duplication.
In the first iteration the user starts by labelling a few relevant and irrelevant samples; this is the same for every method and therefore all lines start from the same point. The size of the training sets of the randomly selected samples, first samples and the user feedback samples (respectively 2, 3 and 4) are the same. Obviously the training sets where all samples are used (1 and 4), with or without super sampling, are larger. The training set constructed from the random sampling and only the first samples are added to compare to the methods that use the user feedback. We want to verify if the samples that the user chooses to label based on the errors of the previous classifier really hold more information and are more valuable than other samples.

In Figure 7.7 we can see that the combination of using all samples including those that the previous classifier labelled and super sampling of the samples labelled by user feedback gave the best results in most iterations and needed the smallest number of iterations to get to a low error rate. All the methods that use the samples labelled by the user feedback get a lower error rate and need less iterations for achieving this. The first iteration this difference is not there yet and the random choosing seemed to have an advantage of spreading the samples and not choosing some not so good samples that made the ‘all samples’ get the worst score in that iteration. The best curve is the method that used all samples available but super sampled the ones gathered through user feedback (black line). We can conclude from this that the user feedback picks the hard to classify sample, which is comparable in some way to the active learning process where the system also presents some samples to the user that are especially picked because these are hard to classify.

Now we want to compare this new way of using the user feedback to the standard way of using all features in the four chosen classifiers from the previous chapters. This is
done in the same scenario as the previous test: in the first iteration the user starts by labelling a few relevant and irrelevant samples; this is the same for every method. The system trains a classifier using these samples labelled by the user in its training set. Afterwards the system shows the user the relevancy plot for the following samples of the DVD based on the labels given by the classifier and the user will adapt the relevancy plot when something is labelled wrongly. Then the classifier is trained again using this new information, this is one training iteration. The user is given six training iterations, in which he/she labels fifteen samples per iteration, to make all results comparable. The results of the classifiers after five training iterations is shown below, this is 44 samples in the first iteration and 4*15 samples user feedback that are labelled.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Error rate</th>
<th>Error rate per event</th>
<th>Precision</th>
<th>Recall</th>
<th>F5 measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal kNNC</td>
<td>0.0198</td>
<td>0.0714</td>
<td>0.9653</td>
<td>0.8299</td>
<td>0.8497</td>
</tr>
<tr>
<td>Normal kNNC + super sampling feedback</td>
<td>0.0343</td>
<td>0.0594</td>
<td>0.9582</td>
<td>0.6836</td>
<td>0.7197</td>
</tr>
<tr>
<td>SVC 'p',1, C+= 10*C- post proc.</td>
<td>0.0666</td>
<td>0.0435</td>
<td>0.6382</td>
<td>0.7582</td>
<td>0.7352</td>
</tr>
<tr>
<td>SVC 'p',1, C+= 10*C- post proc. + super sampling feedback</td>
<td>0.0666</td>
<td>0.0435</td>
<td>0.6382</td>
<td>0.7582</td>
<td>0.7352</td>
</tr>
<tr>
<td>Adapted kNNC, post proc.</td>
<td>0.0882</td>
<td>0.1803</td>
<td>0.5326</td>
<td>0.9015</td>
<td>0.8082</td>
</tr>
<tr>
<td>Adapted kNNC, post proc. + super sampling feedback</td>
<td>0.0225</td>
<td>0.0455</td>
<td>0.8985</td>
<td>0.8716</td>
<td>0.8760</td>
</tr>
<tr>
<td>kNNC, pp. incl ‘uncertain’</td>
<td>0.0163</td>
<td>0.0556</td>
<td>0.9654</td>
<td>0.8480</td>
<td>0.8655</td>
</tr>
<tr>
<td>kNNC, pp. incl ‘uncertain’ + super sampling feedback</td>
<td>0.0199</td>
<td>0.0541</td>
<td>0.9633</td>
<td>0.8055</td>
<td>0.8281</td>
</tr>
</tbody>
</table>

*Table 7.5. Results of the classifiers after five training iterations. All results are per sample; error rate is also calculated per event.*

From the above results after 5 iterations we can not conclude how good the classifiers are because it is only based on the last, largest training set. Therefore a learning curve for the five iterations is plotted for each classifier with and without super sampling of the user feedback. This is shown in Figure 7.5 below, in which the error rate per sample is plotted for the five iterations for every classifier.
In the above figures a few aspects can be noted: the nearest neighbour classifiers seem to profit from the super sampling of the user feedback, noticeable from the lower error rate per iteration for the regular kNNC and the post processed adapted kNNC. For the post processed Support Vector classifier the gain of the super sampling is barely visible and in Table 7.5 was already to be seen that the final results after five iterations were exactly the same. The post processed adapted kNN classifier with the ‘uncertain’ class shows a different behaviour: results without super sampling were better for this classifier, although the final results were very similar after five iterations. The user has in this case adapted the relevance feedback plot to give feedback on the ‘uncertain’ class as well as the completely wrongly labelled samples. The same results can also be calculated per event instead of per sample, which gives a better insight in the results of a classifier in terms of events, like missing a complete applause. This is more important then detecting an applause a bit late. The error rate per event is shown in the following figure.

Figure 7.8. Learning curves of the classifiers with and without super sampling of the user feedback. On the x-axis the iterations and on the y-axis the error rate per sample.
In the above plot it is clear as well that for the most kNN classifiers super sampling does improve the learning curves. The post processed adapted kNN classifier with the ‘uncertain’ class showed better behaviour without super sampling in Figure 7.8 where the error rate was measured per sample. This difference is not present in the error rate per event: the super sampling does not improve nor make the results worse. This behaviour might be caused by the early detection of ‘hard to classify’ samples in the uncertain class. The samples are then labelled by the user feedback but this causes the algorithm to focus on the early samples, especially in the first iterations, and those samples might not be generic enough.

Concluding this paragraph on stressing new samples by super sampling the new user feedback in the training data it can be said that super sampling of the user feedback generally improves the results in two-class nearest neighbour classifiers. The first set of wrongly classified samples every iteration holds more valuable information than random samples or the first samples. This means that the way that the user interacts with the system is efficient. Super sampling this user feedback does stress the ‘hard to classify’ samples which also improves the next classifier.
Chapter 8.

Conclusions and future work

This chapter gives an overview of the project’s contributions. After this overview, we will reflect on the results and draw some conclusions. In Paragraph 8.2 there will be an extensive discussion about the results and the implications. Finally, some ideas for future work will be discussed in Paragraph 8.3.

8.1 Conclusions

We proposed a method for identifying segments of a video that represent user specified events by using interactive learning with no prior information about the video nor the events. This is implemented using music DVDs specifically. Special research questions formulated in this proposal were:

1. What classifier suits this scenario best?
2. Is it viable to make choices concerning the classifier, its parameters and other variables for such a general system with no prior information about the video nor the events?
3. What are the practical differences between this approach and classical, passive learning approaches?
4. What are the practical differences between this approach and the classical relevance feedback approach on image collections?
5. Will such a system converge well enough and in how many iterations?
6. What needs to be adapted to this system in order to apply this in practice to surveillance videos?

These questions will be answered in this chapter according to our findings during the research.
1. What classifier suits this scenario best?

The choice of classifier was made based on the following requirements to the classifier:

1. Capability of learning efficiently from different sizes of training sets.
2. Capability of handling unbalanced datasets.
3. Capability of handling all different sorts of problems: simple, complex, separable and overlapping.
4. Number of parameters to be set (using prior knowledge).
5. Training speed.

It turned out that the classifier that was the most general suited our scenario best. By general we mean a classifier that was able to achieve nice results on different training set sizes and various sorts of problems. We found that this general behaviour was dependent on number of parameters to be set, so the less parameters the more general a classifier could behave. The capability of handling unbalanced datasets has been solved for the classifiers tested by adapting the C parameter in the SVC, adapting the nearest neighbour algorithm using class priors. The logistic linear classifier did not have a lot of problems with unbalanced datasets but showed the least results on the smallest train sets. Training speed has been the least important requirement because our focus was more on the results, and on the datasets used the training speed has been limited to < 60 seconds which is reasonable for a user to wait. Only optimizing the kernel parameters of a ‘rbf’ SVC turned out to be too time consuming as cross-validation was used and therefore this method has not been chosen. Classification speed will be more important in a real-time application.

The classifier that turned out to most applicable in practice for every event, DVD and training set sizes was the nearest neighbour classifier, adapted or post processed to cope with the unbalanced dataset.

2. Is it viable to make choices concerning the classifier, its parameters and other variables for such a general system with no prior information about the video nor the events?

It turned out to be very hard to make choices concerning the classifier, its parameters and other variables for such a general system and this was especially clear when Support Vector classifiers were used.

The choice for the adapted nearest neighbour classifier has been based on training and testing on the music DVDs in the specific scenarios. Some ‘prior knowledge’ about the DVDs and events has been used, for instance the minimum length of the user specified events and the choice of the features. These choices have been necessary to make the system work properly. This can be justified by claiming that the ‘prior knowledge’ has been very general for music applications and could also be based on literature instead of the data. It should only limit the system to music data.

Choices concerning classifier parameters have been made offline in the Support Vector classifier and it turned out this is not realistic. We have chosen a polynomial kernel after finding that the radial basis kernel had problems finding a solution and if found it was highly over trained. The results of the SVC were comparable or sometimes better if the corresponding kernel parameter(s) were optimized online regarding the (train) data in the within DVD scenario, but no better or even worse in the across DVD scenario. Without optimizing the parameters the results of the SVC were unstable: sometimes
specifically poor, sometimes good and sometimes the SVC found no solution at all. The choice of the kernel and its parameters were crucial and problem dependent. It is already known in pattern recognition that a Support Vector classifier can yield very good results if the kernel and parameters are optimized with respect to the data. This brings us to the conclusion that for such a general system the only requirement to a classifier is to be simple. Of course this will limit the system to simpler problems but more complex problems that need more complex classifiers need more effort in parameter tuning, which is impossible in our scenario.

3. **What are the practical differences between this approach and classical, passive learning approaches?**

There are three large differences. The first two differences are differences in the requirements for the classifier that are imposed by the scenario.

1. No prior knowledge is known about the data or events which means no optimization of parameters and testing on separate independent test set is possible. This makes it harder than classical learning.

2. The iterative learning behaviour demands a flexible classifier that can handle both large and small training sets and different types of problems in terms of complexity.

3. The scenario also gives the opportunity to learn from the interaction: new samples that are added are more important just because they are newer. But more important: a user will only give feedback when a classifier labels something wrongly. This resembles active learning in a way because only hard to classify samples are labelled by the user and added to the training set. Since these examples are around the classification boundary the number of examples to learn a concept can often be much lower than the number required in normal supervised learning. In active learning though, there is a risk that the algorithm might focus on unimportant or even invalid examples since the learner chooses the examples [27]. This is not an issue in our case because the user picks the examples.

4. **What are the practical differences between this approach and the classical relevance feedback approach on image collections?**

The classical relevance feedback scenario is different from our scenario because of the ordering of the samples based on the extra dimensionality: time. This has two major implications: we can not look forward in time so there is no pool of unlabelled samples. This prohibits a classical active learning approach. It also implies that a ranking of the most relevant or similar samples as is done in classical image retrieval is not possible. Our system does not return a ranking but on every video frame a user watches it gives an indication of its relevance. These differences are especially relevant in the surveillance applications as the main goal of such is system is to raise an alarm when an alarming situation occurs in real-time. If the data would be available beforehand, which could be the case when browsing your own music DVDs, the upcoming frames until the end of the video can be regarded as a pool of unlabelled samples. This is a scenario that is quite similar to example-based image retrieval. Samples are then two seconds of video instead of an image. A lot of dependencies would be present in the dataset.

5. **Will such a system converge well enough and in how many iterations?**

This is dependent on the events chosen by the user. Simple event choices like applause are quite easy to distinguish and our classifiers have shown that an error rate of 5% is
reachable within 2 to three iterations of 15 samples. Because of the overlap, separating guitar from piano has not given satisfactory results in our scenario with the limited amount of samples available.

6. What needs to be adapted to this system in order to apply this in practice to surveillance videos?

First of all the features need to be changed: there is a need for a general set of global and local video features. There is a need for low-level video feature extractors that do not make any assumption about the target events nor about the number and configuration of moving objects of interest that users will request. This is what Meessen et al. propose in their interactive smart surveillance system [22] but this approach needs pre-processing and is not suitable for real-time applications. Another set of features that might suit our purpose well are the 3-dimensional SIFT features proposed by Scovanner et al. [34] that extended the fast and robust Scale Invariant Feature Transform (SIFT) algorithm with a temporal dimension. The video feature set could also be inspired by the new TRECVID [37] event detection in surveillance video assignment, which is a pilot in 2008 and might results in a good general video features dataset for surveillance purposes. The audio features can be used in addition to these video features when audio is available in the surveillance video. Such a multimodal analysis method for semantic understanding of video must include a (early or late) fusion step to combine the results of audio and video feature analysis [40].

Sub sampling of the data will be necessary as the dataset will get even more unbalanced than the music DVD datasets. Another reason for sub sampling is that a surveillance feed can be seen as a dataset that increases to infinity and no system will be able to memorize and use all the data. This is especially important for when the chosen kNN classifier will be used as this needs to memorize all samples and classification time is based on the number of distance evaluations to all these points. This could be optimized by partitioning the feature space, and only computing distances within specific nearby volumes using algorithms like k-dimensional trees (kd-trees, [3]) or locality sensitive hashing [16] but this optimization will not completely solve the problem of the increasing training set. Therefore sub sampling is not only necessary for balancing the dataset a little, but even more important for satisfying memory constraints and keeping the classification time low. A good implementation for this type of super sampling is basing it on the age of the data such that old (non relevant) samples are thrown away at a higher rate than new samples.

A drawback of surveillance videos is that still a few different relevant events have to be labelled, which in surveillance sequences are very rare. An option is to do the first training part on an offline database in which there can be searched for known events to establish a stable basis for the system and afterwards the system can still be adapted to current circumstances. Another option is to attach the system to an already working system such that when somehow action is undertaken in response to the video an automatic signal will label the current video part as being relevant. Besides this drawback about surveillance data, there is also an advantage when the system will be adapted to surveillance videos: because of the static behaviour of the one camera used and the overload of negative samples a very reasonable ‘best educated guess’ can be made. A smart surveillance system would be trained per camera and as the viewpoint of this camera is static the generalization performance of our system is not directly
important: comparable with only within DVD behaviour in the music DVD scenario. Because of these characteristics of the surveillance dataset a one-class classifier needs to be investigated as well.

This smart surveillance system can be developed and tested on the new TREC Vid [37] challenge ‘surveillance event detection’ Gatwick Airport surveillance video data provided by the UK Home Office Scientific Development Branch. It comprises about 100 hours - the output of 5 cameras from the same period of 20 hours (2 hours per day over 10 days). It is good to have such a large and standard dataset available for these purposes because especially annotated surveillance data is often very hard to get and now there is a platform to compare systems with each other.

8.2 Discussion

After answering the main research questions in the previous sections based on the findings of the literature research and our experiments there are still a few items left to discuss. These will be discussed in this paragraph and are related to the following topics: using prior information in the scenario, SVC behaviour, interactivity design choices, user testing and generalizing behaviour of the classifiers in the across DVD scenario and the dependence on the data distribution.

Using prior information

The experiments have shown that interactive learning for video content analysis can be applied on many different levels of interaction. We have chosen to let a user watch music DVDs and interactively adapt a curve that represents the importance of the video frames. This way the user helps labelling a training set for a classifier. When a classifier is chosen to process the relevance feedback a few problems appear, that are mostly related to the fact that no knowledge is available about the data. First of all, the ‘general audio feature set’ that needs to contain information about all possible events and videos is estimated based on literature reviews of audio features and can not be tested and selected based on our data. Another discussion point is the feature ranking based on the possible events that we did in Paragraph 3.3 to achieve a starting feature set that is not too correlated and also useful for our purpose. We do not know these events in the real application, so there should be another way of achieving this feature set. The problem with not using the labels –for example using PCA - is that weak features are favoured because there is no correlation with the good features. This all comes back to some sort of circular dependent problem: we want good features but can only define good features in terms of events, which we do not know beforehand. Secondly, the preferred normalization methods need the mean and variance of the features of the complete video. Everything that is or can be optimized depends on the data and events - classifier choice, classifier parameters, sub sampling training data, post-processing steps etc. We have done the optimization prior to the interaction but this is not possible in for instance a real-time surveillance scenario. In a streaming surveillance scenario no prior information about the data is available and needs to be estimated from the data available at that point, but there is some prior knowledge about possible relevant events. This is not the case per se in the music DVD scenario as the complete DVD is available and the
only uncertainty is the likings of the user. On the other hand, in the music DVD scenario the data is often available but the user is completely free to choose any event. This would also imply that the results could be a little better as we can for instance lose the delay caused by only classifying samples based on the previously seen samples in the music DVD scenario. The restrictions we decided upon could have been loosened if the application and scenario choices would have been chosen in an earlier stage instead of being too general.

**SVC behaviour**
The main problem we have found was finding an appropriate classifier without any prior knowledge and especially finding the parameters. Parameter tuning turned out to be especially hard for the Support Vector classifiers: the kernel parameter turned out to be crucial and the default value gave inferior results. Optimizing this parameter using the (small) available training set requires cross validation and testing and this turned out to cost too much time for practical reasons. Another drawback of this optimization is that the algorithms might overfit easily and result in poor generalization performance. KNN classifiers showed better performance. This can be explained by the fact that this simple classifier needs no offline parameter optimization. The disappointing results for the Support Vector classifier, that is a very popular classifier and has been used lot with practical success in various areas, can be explained by the lack of heavy parameter tuning normally performed for these classifiers. But the optimized version of the SVC did not yield significantly better results either: this can be explained by the fact that the SVC started to behave more or less like a memory-based classifier like the kNN classifier when the dataset was very unbalanced. This can be seen by the number of positive support vectors that was over 90% of the positive class, a behaviour also recognized by Yang and Hauptmann [46]. They have recently shown that the SVC tends to behave like a nearest neighbour classifier with a large \( k \) when the concepts to be learned are hard and the training set unbalanced. They have evaluated the generalizability of the classifiers used for video concept detection and especially compared the commonly used Support Vector classifiers with kNN classifiers. The TRECVID [37] datasets were used for their testing. They observed that when the training set was unbalanced the SVCs in general learned little from the data besides memorizing the positive samples, because over 90% of the positive train samples were support vectors (SVs). Therefore the SVC started to behave like a kNN classifier only it makes predictions based on SVs instead of nearest neighbours. The real problem might be that the features were not general enough or the classification algorithms just could not grasp the essential patters of the concepts. In our case we have seen very comparable behaviour: around 30-80% of the positive samples were SVs and only 3% of the negative samples. The numbers are a bit lower in our case probably because of the overlap in the features.

**Interactivity choices**
Also choices concerning the design of the interactivity with the user can be discussed. There are more ways to implement the feedback in the interface, e.g. dragging/clicking on graph to get a size \( k \) window, or to make the system estimate the best \( k \) by selecting ‘similar’ feature vectors around the selected point (clustering). Although this last idea has a problem that not all features are important so similarity is hard to measure, we need to know which features are important for this but to find out we need the window – chicken and egg problem.
Besides events, also the videos that we have tested on should be free to choose by the user. This means a user could select any event from any video to be relevant. This concept causes the problem set on which our system should work on to be infinitely large. It is already clear that none of the existing classifiers can currently separate all problems: some classes are just not separable or linearly separable; others need very specific tuning of a specific classifier. This is still a problem to be solved. There is no general problem solver (classifier) in classical machine learning. The problems we try to solve are the same as classical learning problems and we use “classical” classifiers in our interactive systems, so this problem holds for our scenario too.

**Generalizing behaviour**

The applause versus singing dataset seemed to be a little easier to classify than the applause versus solo in the ‘inter DVD’ scenario. We expected at first that separating solos from applause would be more easy because of the large difference in ‘noisiness’, but it turned out that applause versus singing gave better results. We expect that this is caused by the different instruments that are used in the solos and therefore this is a more complex distributed dataset. These are not that hard to separate from applause but when it comes to generalization the class of solos is probably more complex then the singing. Therefore the test DVD solo samples can be quite different in feature space from the train samples and more DVDs are needed in order to get the same cross validation results.

### 8.3 Future work

In the previous two sections a lot of ideas for future work have been mentioned as a cure for solving some problems and in adapting the system to the surveillance scenario. Because these ideas have been discussed quite extensively they will be shortly summarised in this last section.

The temporal integration method we have adopted is taking some statistical measures of the features. More sophisticated feature integration models have been proposed, such as the MAR model proposed by Meng et al in 2007 [24]. Implementing this new method for temporal feature integration might allow us to generate a better general audio feature set. This audio feature set can be combined with a good general video feature set, such as the 3-dimensional SIFT features proposed by Scovanner [34]. Such a multimodal analysis method for semantic understanding of video must include a (early or late) fusion step to combine audio and video feature analysis [40].

Convergence of the system was one of the research questions. This is not surprisingly very dependent on the videos and events chosen, but could also be dependent on how the user picks the samples. Does the user pick only the correct samples or are there some samples that are on the edge of different and possibly overlapping events? And another factor is the number of samples a user labels per iteration and how specific the user is in adapting the relevance curve. A future research step to improve the interactivity is to do user testing to see how the interaction of a user with such a system really takes place.
This will not only probably result in a better algorithm, but also improve the graphical user interface and usability of the system.

To apply this interactive system to a surveillance scenario a couple of changes are made in the scenario. Because of these changes in the scenario either strictly only past samples can be used in training and pre and post-processing or a training stage should be incorporated using an offline database. A ‘best educated guess’ can be implemented based on the overload of -unlabelled and most probably negative- samples available. Besides changes in the scenario there are also some adaptations necessary to cope with the different type of dataset: a general set of video features should be developed and combined with audio features; sub sampling needs to be performed in order to let the dataset not approach infinity and a one-class classifier should be tested.


[21] M.A. Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. *Workshop on Learning from Imbalanced Data Sets II*, ICML, Washington DC, 2003


[26] S.U. Naci and A. Hanjalic, Content-Based Indexing of Music Concert Recordings Based on Crossing-Rate Features, CBMI 2008


[49] www.fabchannel.com

[50] www.last.fm