Faculty of Media Engineering and Technology
German University in Cairo

Multimedia Analysis and Data Mining Group
Deutsches Forschungszentrum Für Künstliche Intelligenz GmbH

Audio Features for Automatic Video Tagging

Bachelor Thesis

Author:	Dalia El Badawi
Reviewer:	Prof. Dr. Andreas Dengel
Supervisors:	Dr. Adrian Ulges
	Damian Borth
Submission Date:	3 September, 2011

Faculty of Media Engineering and Technology
German University in Cairo

Multimedia Analysis and Data Mining Group
Deutsches Forschungszentrum Für Künstliche Intelligenz GmbH

Audio Features for Automatic Video Tagging

Bachelor Thesis

Author:	Dalia El Badawi
Reviewer:	Prof. Dr. Andreas Dengel
Supervisors:	Dr. Adrian Ulges
	Damian Borth
Submission Date:	3 September, 2011

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowlegement has been made in the text to all other material used

Dalia El Badawi
3 September, 2011

Acknowledgments

This thesis resulted from my research work at the German Center for Artificial Intelligence (DFKI GmbH) in Kaiserslautern, Germany. So first, I would like to thank Prof. Dr. Slim Abdennadher and Prof. Dr. Andreas Dengel for providing such an opportunity. And many thanks to Dr. Thomas Kieninger and Jane Bensch for handling the administrative issues and all the necessary paper work.

Then, I would like to show my appreciation to all those who have helped me in one way or the other during the completion of this project. I wish to express my gratitude towards my supervisors: Dr. Adrian Ulges for his help and support, and Damian Borth for his help through out my work at DFKI and for his guidance on writing this thesis.

I also wish to acknowledge the moral support of my fellow GUCians (you know yourselves) who kept me company in the HiWi room and maintained a (nearly) constant supply of (somewhat) positive energy and chitchat that I now can not imagine having to complete my project without.

Last but not least, I sincerely thank my family who continually provide me with the utmost support in all my endeavors, and special thanks to Hassan for his help in proofreading this thesis.

Abstract

While the majority of techniques for automatic video tagging mainly consider the visual aspect of the video, this thesis addresses the analysis of the acoustical counterpart–the video soundtrack– for learning concepts. Consequently, the goal of this thesis is to explore audio features for the task of automatic concept detection and measure the improvements added when used along with visual features. Additionally, the thesis briefly investigates the temporal correlations between audio and visual modalities. Experiments conducted on real-world data from YouTube showed that a mean average precision of 0.47 could be achieved by audio features alone and increased to 0.62 by a combination of audio and visual features, an improvement of more than 7% over a pure visual system.

Contents

Chapter 1

Introduction

1.1 Motivation

Computers and recording equipment for both audio and video have become ubiquitous (e.g. built-in in cell phones) and are getting cheaper, giving more people the ability to record, edit, and upload their content. This is especially valid for videos on popular sharing websites such as YouTube [2], which has created a massive database of videos covering diverse subjects and concepts. The need to efficiently search and navigate through the database is thus necessary.

Usually videos have associated metadata that describe them such as title and date. Searching and retrieval is then done by matching a search query to the metadata. However, a lot of videos are either unlabeled or improperly labeled. In addition, searching by using only the labels might return a lot of irrelevant results since labels may be generic (e.g. music video) and apply to a multitude of videos leaving the user unsatisfied and wasting time manually searching for the desired videos.

Therefore, appropriate labels or tags are needed. Those tags could be based on the semantic concept of the video, where a concept is an object or event occurring in the video. Consequently, concept detection is required. Some concepts are better detected visually like sunsets, some acoustically like cheering, and some can be detected both ways like cats for instance. But given the massive database described earlier, hiring people to watch and tag videos would not only be time consuming but also expensive. To this end, the objective is to automatically detect concepts.

In order to automatically detect concepts, descriptors or features that describe the actual content of the video are desired. These could be visual features if they describe what is seen or audio features if they describe what is heard. The challenge is finding robust features, meaning features that are invariant for videos belonging to the same concept and different between videos belonging to other concepts. For example, the features describing a video of a cat should be consistent for cats of different colors and sizes and in different lighting conditions but very distinct from videos of dogs. Similarly for audio, the meows of different cats and in different surrounding environments should have consistent features. To that end, robust features– audio and visual– are needed.

Besides having robust features, there is the classification task which generates a model representing the learned concepts. A robust classifier is one that, given features, produces a model with enough discrimination power to separate between the different concepts. Finding such a classifier is yet another challenge that faces researchers mainly due to the statistical variability and the high potentiality of present noise in the available data, which makes the discrimination between the different concepts difficult.

Feature extraction and classification techniques has been researched for various tasks and purposes. TRECVid [23] is a conference series aimed at encouraging research in video retrieval which includes advances in feature extraction. It provides standards that allow comparison of different approaches in the field. Those include a corpus of test data and scoring methods. While most research has focused on the visual aspect of the video, there exists applications where audio features supersede visual ones. One such application is identifying multiple videos of the same event [7]. Whereas visually, the same event can be captured from different angles where there might be occlusions, the surrounding audio should be the same even with added noise [7].

Due to the fact that audio and visual modalities tend to be related (e.g. music and dancing), using audio and visual features together is reasonable. And audio features which seem to be helpful for certain concepts are gaining popularity in the research community and are being used in conjunction with visual features to aid the task at hand. Some use audio features independent of visual features, while others try to extract them simultaneously with visual features. This audiovisual relation was, for example, exploited for speech recognition [15], where geometric visual features from the speaker's lips were taken into account along with the audio features to improve recognition accuracy in noisy environments.

Automatic concept detection opens the doors for numerous applications. Other than automatic video tagging, popular applications include content-based retrieval, providing recommendations to users based on the concepts they watch or listen to, and eliminating spam videos where labels (metadata) do not correspond to the content of the video. Specifically for audio concept detection, applications include auditory scene analysis, speech processing, music genre classification, and automatic sports highlights creation. For instance, goals in a soccer match [8] can be detected based on the excitement heard from the commentator or audience, and a clip of all goals can be generated automatically without the need for manual look up.

To sum up, for automatic video tagging, concept detection is needed which in turn requires feature extraction. This thesis is concerned with the feature extraction part, audio features in particular. The audio features are evaluated separately and in conjunction with visual features to measure the enhancements added by audio. In addition, the temporal correlations between audio and visual features are briefly examined.

1.2 Thesis Outline

The thesis is organized as follows: Chapter 2 provides basic principles from concept detection and machine learning. Chapter 3 gives an overview of related literature that uses audio features or a combination of audio and visual features for different tasks. Basics of audio and the approach to audio features as well as audiovisual fusion are presented in Chapter 4. Then, all experiments testing audio features and audiovisual fusion and their results are presented in Chapter 5. And finally, a conclusion of the current work and further suggestions are given in Chapter 6.

Chapter 2

Background on Concept Detection

This chapter defines some basics and terms related to the field of machine learning and more specifically concept detection.

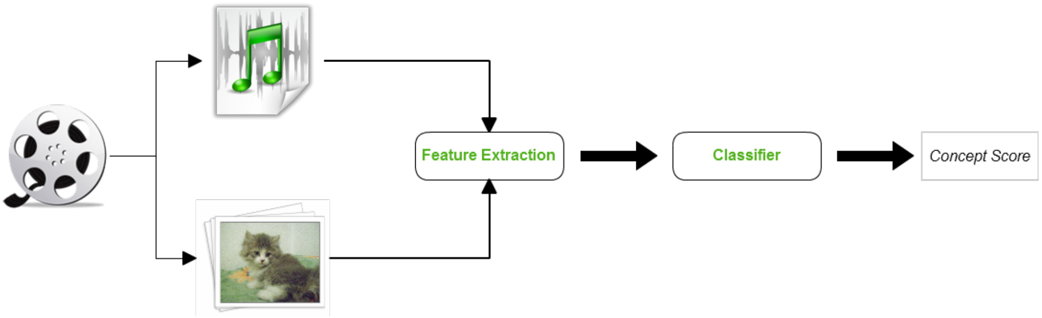


Figure 2.1: A generic multimedia concept detection system. The input is a multimedia file. The audio and/or video streams are used for feature extraction. Finally, the classifier uses the feature vectors to output a score indicating how likely the file belongs to a concept.

Figure 2.1 shows an overview of a multimedia concept detection system. The goal of such a system is to predict the most likely concept for a given input file. Here, the input to the system is a multimedia file containing two media types: audio and video. Then, the required types are extracted for analysis. Afterwards, features or descriptors are computed, either for each type separately or for several types synchronously. The output is a set of the so called feature vectors. The next step is quantization or parameterization of the feature vectors where the entire file needs to be represented by one feature vector. Many techniques exist to do this; one example is to calculate the mean of all feature vectors. Now the set of vectors, one per file, are used to train a classifier to be able to distinguish between the different concepts. The classifier takes all inputs and creates a predictor function which then outputs a score (or probability), for a given query file, indicating the most probable concept. The following sections explain the different components of the system in more details.

2.1 Multimedia Containers

A multimedia container is a file that combines several media types in one file, most typically a video stream and an audio stream. It may also include subtitles or metadata for instance. The format of a container defines its structure and/or compression type. Common formats include Flash Video (FLV) and Audio Video Interleave (AVI). Each stream is, however, coded separately. For example, audio can be stored in a variety of formats such as WAV and MP3; images for the video can be stored as JPEG. The format for each stream defines the decoding scheme needed to understand it; therefore, each type needs a separate decoder to translate it into meaningful sounds in the case of audio or colors in the case of video. An uncompressed format or the stream after decoding is called raw. The feature extraction described next most often works on raw media.

2.2 Feature Extraction

Not only is audio and video content very computationally expensive to analyze in terms of memory and processing time, but also there is really no need to use all the information present for differentiating between different files. Therefore, it is very reasonable and efficient to find compact ways of describing the content and discard information that might not be useful for discrimination. Feature extraction provides the solution by reducing the data to *features* while retaining the discrimination ability. Features are basically a mathematical quantitative representation that describe the content in a compact form. And they are chosen in such a way that they are invariant for videos belonging to the same concept but different for different concepts.

In order to arrive at a compact representation that also carries the most important information required for classification, the features might be designed to mimic humans' perception, given humans' ability to recognize and classify their surroundings, a property that has been exercised in lossy compression techniques. In audio compression for example, psychoacoustical phenomena such as masking (when two frequencies are close enough in time but one has a much higher amplitude, it masks the other frequency; it becomes inaudible) or the hearing range are exploited to discard some information. Similarly for chroma subsampling in image compression which makes use of the human eyes' higher sensitivity to brightness than colors to discard some of the color information.

2.2.1 Audio Features

As indicated by their name, audio features are those extracted from an audio signal. They encompass some characteristics of the signal such as timbre or pitch. They may be divided into three categories: temporal, spectral, and psychoacoustical, depending on the format of the signal when the features were extracted or what they try to describe. Common features include the Zero Crossing Rate (ZCR) which is the number of times the signal passes the zero line, the Mel Frequency Cepstral Coefficients (MFCC) which describe the spectral shape of a signal but also consider some psychoacoustical properties of the human hearing, and pitch which describes the perceived frequency. Usually a combination of features is used. In addition, the time derivatives (typically first and second) along the signal, the mean, variance or other statistical values are also used. More details on audio and audio features are covered in Chapter 4.

2.2.2 Visual Features

Similarly, visual features are those extracted from the frames (images) of a video. They capture characteristics like Color or Edges. But unlike audio where features are extracted from the entire signal, visual features are only extracted from selected keyframes. This is mainly due to the high computational requirements as well natural redundancies in the frames which are needed for the appearance of smooth motion but would not supply significantly different information that would help in discrimination between different concepts. Keyframes could be sampled at regular intervals or some algorithm may be utilized to decrease the number of keyframes extracted for efficiency and computational reasons. Visual features can be divided into global and local features. Global features describe the overall content of the image and thus cannot separate foreground from background. An example is the Color histogram which describes the distribution of colors in the image. Local features describe parts of the image and could therefore be used to discriminate between the different objects present [19]. The following describes the Scale Invariant Feature Transform (SIFT) which is a popular local visual feature.

SIFT

As its name implies, this feature is not affected by transformations that could take place in an image such as scaling or rotation in addition to illumination and viewpoint. The algorithm starts by identifying keypoints like corners over all scales and locations of an image. Gradients are calculated around those keypoints and are then transformed into a 128-dimensional vector representation. In the end, an image is represented by a set of 128-dimensional vectors. A detailed description of SIFT can be found in [18]. There is also the dense version of SIFT, where instead of generating descriptors at certain keypoints, they are generated at points sampled from a regular grid over the image.

2.3 Machine Learning

Machine learning is process of observing data and trying to find the underlying model so that the machine can predict new class labels or sample points based on what has been observed. For classification, there are two types of learning algorithms: the supervised methods and the unsupervised methods. The former requires the input data to be labeled where each sample is associated (labeled) with a certain class and we speak of classifiers. The latter, however, does not require any labels and just learns associations between the data samples and we speak of clustering. Classifiers and clustering as well as a few more concepts are explained next.

2.3.1 Distance Metrics

The similarity or dissimilarity between data samples needs to be quantified so that a classifier can do its job. Distance metrics serve that purpose. Since samples are ultimately just n-dimensional vectors, a function $d(X, Y)$ can be defined to measure the distance between two vectors X and Y. Many distance metrics are available such as the Euclidean Distance and the χ^2 distance. Given two n-dimensional vectors $X = [x_i, i = 0, \ldots, n-1]^T$ and $Y = [y_i, i = 0, \ldots, n-1]^T$, the Euclidean distance d is:

$$d(X, Y) = \sqrt{\sum_{i=0}^{n-1} (x_i - y_i)^2} \tag{2.1}$$

and the χ^2 distance is:

$$\chi^2(X,Y) = \sum_{i=0}^{n-1} \frac{(x_i - y_i)^2}{x_i + y_i} \tag{2.2}$$

where if x_i and y_i are zero, then the term is zero.

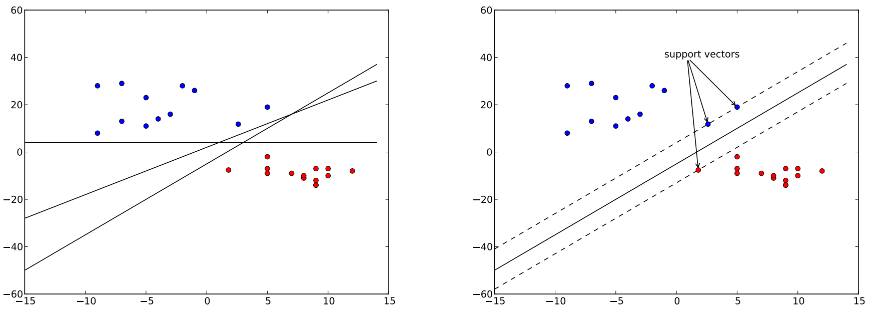
2.3.2 Classifiers

Classifiers are algorithms used to learn to distinguish between classes of data. They are one type of learning algorithms that require labeled data for training. For example, the Nearest Neighbor rule is a simple classifier which compares a query sample to the rest of the data and assigns it to the same class as its best match. The comparison is done using a distance metric like those described in Section 2.3.1. For the training of a classifier, the entire dataset could be used except for one sample which is used for testing. This is called leave-one-out method. Alternatively, the dataset could be divided into a training set and a testing set. Described next is another prominent classifier known as the Support Vector Machine (SVM).

SVM

SVM is a binary classifier which, given the training data divided into two groups labeled either 1 for samples belonging to the first class or -1 for samples that belong to the other class, predicts a score to an unknown sample indicating which of the two classes is more probable. Using the training data, it finds the optimal hyperplane that gives the largest margin between the two classes as illustrated in Figure 2.2. If data is linearly separable, then there exists a unique global optimum unlike other classifiers that might be trapped with a local optimum. The points closest to the margin that define the hyperplane are known as support vectors. The further away a sample is from the hyperplane, the more confident the SVM is with its prediction. One issue that can arise if the generated hyperplane is too complex and uses too many support vectors is over fitting, which means that the model is tailored for the training data and would fail to generalize with the testing data. In order to overcome this phenomenon, a cost parameter C is used to allow some misclassification, where the higher the value the more accurate the model represents the training data and the more prone it is to over fitting. Generally, there is a trade-off between misclassification and the SVM generalization ability. C can be optimized using a grid search.

Moreover, since not all data is linearly separable (in their original space), the kernel trick is used. That is a mapping of the data to a higher dimension



(a)

(b)

Figure 2.2: a) Several possible hyperplanes can separate between the two classes. b) The SVM finds the hyperplane that provides the largest distance between the two classes.

where linear separability is possible. The mapping is done using a kernel function such as the Chi Square Kernel: .

$$K(X, Y) = \exp \frac{-\chi^2(X, Y)}{\gamma} \tag{2.3}$$

where χ^2 is the distance function and γ a tunable parameter.

Furthermore, in order to use a binary classifier for a classification problem with more than two classes, two methods exist, namely one vs one and one vs all. One vs one means training SVMs to distinguish between each pair of classes. For c classes, $c(c-1)/2$ classifiers need to be trained. Classification is then based on the number of times a class was assigned. While one vs all means training SVMs to distinguish between one class against the rest, which only requires c classifiers for c classes. In this case, classification is based on the highest score. Once the hyperplane is obtained, the classifier can now do predictions on new unseen data points. Finally, the score of the SVM is the signed distance to the hyperplane. A sigmoid function can be fitted so that the distance can be transformed into a probability.

SVM has several advantageous properties like the good generalization ability even when only a small number of training data is available, in addition to its ability to handle high dimensional data. For more details on the theory of SVM, please refer to [24].

2.3.3 Clustering

Clustering is an unsupervised machine learning technique where points that are similar according to some distance metric are clustered (grouped) together. One popular algorithm for clustering is the k-means algorithm. k-means starts by randomly choosing k points to be the means of their respective future clusters. It then goes on with creating the clusters by finding the closest points, using a distance metric (typically the euclidean distance), to those means. Due to the random start, different initial chosen points would result in different clusters.

2.3.4 Evaluation Measures

Evaluation measures are used to quantify the performance of a classifier. The standard measures used by TRECVid [23] are Average Precision (AP) and Mean Average Precision (MAP). Precision is the ratio of the number of correctly classified videos (true positives) to the total number of videos classified as positive whether right (true positives) or wrong (false positives). In other words, precision measures the probability that videos are correctly classified. For a ranked list of N test videos based on the classifier scores, the precision at each rank from top to bottom can be calculated. Then AP is the average:

$$AP = \frac{\sum_{r=1}^{N} P(r) \times T(r)}{number\ of\ true\ positives} \tag{2.4}$$

where $P(r)$ is the precision at rank r and $T(r)$ is a binary function indicating whether the video is a true positive. AP rewards true positives and penalizes false positives ranked high in the list. And MAP is the mean over all classes in the dataset. The classifier's MAP is typically expected to exceed random guessing which is the number of positive samples per concept to the total number of samples.

2.3.5 K-fold Cross Validation

Cross validation is a way of utilizing the small amount of available data for training in order to tune system parameters and find the best combination that would optimize performance.

In k-fold cross validation, the training data set is divided into k parts where k-1 parts are used for training the classifier and 1 part is used for validation. This process (training/testing) is repeated k times such that each part would be used once for testing and k-1 times for training.

First, the parameters are tuned using only the subsets from the training set. Then, the classifier is retrained using the entire training set and the optimized parameters.

2.3.6 Vector Quantization

Vector quantization maps data onto a smaller predefined set. This means that each input vector is compared to the vectors in the set and is replaced by its best match: the closest one according to some distance metric. This process further decreases the dimension depending on the size of the predefined set. Described next is one method for vector quantization known as the Bag-of-X.

Bag-of-X Representation

Due to the varying length of videos, the resulting sets of feature vectors do not all have the same size; longer videos have more feature vectors. And classifiers require their inputs to be of the same dimension. Bag-of-X would not only quantize the feature vectors but also give all videos the same dimensionality. This method was first used for documents, where the words were simply counted and a histogram was generated. Analogously, for audio and images, bag-of-audio-words and bag-of-visual-words can be created according to the following steps. First, feature vectors from the dataset are clustered. All feature vectors in the dataset could be used or only a subsample, a random subset. The clustering can be done using the k-means algorithm creating a codebook which consists of all the means of the clusters. These are the codewords. Then, this codebook is used to quantize all vectors so that the inputs are only represented using codewords from the codebook. Finally, a histogram of the count of words is generated, this is the bag-of-X-words. For audio features, the bag-of-audio-words represents the entire video. While for visual features, the bag-of-visual-words represents a keyframe.

It should be noted though that for audio and video (images), the Bag-of-X is a result of a clustering algorithm and not a natural word count as in the bag-of-words for documents. This means that there are parameters such as the codebook size that affect the output.

Chapter 3

Related Work

The first section gives an overview of related work using audio features. While the second section gives an overview of related work combining both audio and visual features.

3.1 Audio

Audio features have been used for plenty of different applications and tasks. The following section discusses audio features and experimental setups used in a number of applications including concept detection.

MFCC are a widely used audio feature that have become a standard in the field of speech recognition and are now being also used for general audio classification. In [28], MFCC were used for determining voice intensity. The training set was divided into three classes: silent speech, normal speech, and very loud speech. For classifying samples from the test set, the k-nearest neighbors classifier was used. The MFCC for each test sample were compared to the training samples and assigned the class of the largest members.

Since MFCC have mostly been used for application on speech segments, it was necessary to examine their appropriateness for music and audio in general. The difference between speech and music lies in the dynamic range (range of frequencies found in the signal) which is wider for music and therefore may contain the bulk of information in the higher frequencies. MFCC have been shown to be appropriate for music modeling in [17]. Two of the steps for calculating MFCC that might have been tailored for speech were examined in a speech/music discrimination task. The first was using the Mel Scale which was compared to a Linear Scale. The Mel Scale performed significantly better than the linear in terms of error rate. Although that does not prove that the Mel scale is good for modeling music, it shows that it is still better than using a linear scale. The other step examined was the final transformation step using the Discrete Cosine Transform (DCT) which is supposed to decorrelate the MFCC vectors. The author found that the resulting vectors were indeed decorrelated and concluded that DCT is appropriate.

MFCC, its first derivative, short-time average energy and a few more audio features were used for audio-based context recognition in [9], where context recognition is basically audio classification in a limited number of classes that determine the surroundings. The dataset consisted of specially made recordings covering different contexts. For classification, k-nearest neighbors and Hidden Markov Models (HMM) were used. In order to compare between the features, the two classifiers were trained using the different features separately. The MFCC and HMM combination gave the highest recognition rate at 63%.

Muscle Fish [26]–an audio classification and retrieval system– used a dataset of audio samples of only singular sounds. Several perceptual features such as loudness, pitch, brightness, bandwidth, and harmonicity were used. Then, a nearest neighbor classifier was used to classify a query sound.

Energy and MFCC were used in [10] for the task of audio and music content-based retrieval. The energy was concatenated with 12 MFCC to form a 13-dimensional feature vector. After feature extraction, the resulting vectors were quantized using a Quantization Tree. The Quantization Tree is argued to be able to handle very high dimensional data because only one dimension is considered at each node during the tree construction. This allowed for a sequence of consecutive MFCC vectors to be concatenated into one supervector in order to incorporate time. The tree was constructed using a supervector of 5 feature vectors which spans a time of 10 milliseconds. Two datasets were used. One consisted of simple sounds to test audio retrieval and the other contained musical excerpts to test music retrieval. The best MAP achieved for retrieving simple sounds was 0.772 while the best AP for retrieving music was 0.4.

Both [26] and [10] retrieve or classify a given query sound. Also [7] falls into this category since a matching was performed between two sounds (to check if they belonged to the same video). However, their presented query-by-example approach might not always be convenient since a user will not always have a similar sound to what they are searching for at hand. In the following papers, semantic retrieval is used which requires a mapping from a text space to the audio feature space.

Ranking of retrieved sounds was tackled by [21]. For each sound, several Stabilized Auditory Images (SAI) were formed. SAI tries to model the auditory system and has higher temporal resolution compared to MFCC which has better spectral resolution. Each SAI frame box was converted into a sparse code using either vector quantization or matching pursuit. Then, the vectors from all boxes were concatenated. After that, the vectors from all frames were added to represent the entire sound file. The dataset consisted of sound effects where each file contained only one sound. Through experiments, SAI was compared to MFCC and was found to be consistently better in a precision-at-top-k curve, though the improvements were not relatively higher. The best results achieved an AP of 0.35. In addition, vector quantization was found to be better than matching pursuit.

SVM classifiers were employed for audio classification and retrieval in [12]. The audio features used were total spectrum power, subband powers, brightness, bandwidth, pitch frequency, and MFCC. Two sets of features were used: the first were all the features except MFCC and the second contained only MFCC. Each file was represented by the means and standard deviations of all its feature vectors. Then, experiments were carried out using the sets separately and together. For multiclass classification, the authors proposed a bottom-up binary tree in which each pair of classes are compared and the winner is promoted to a higher level in the tree until one class wins at the end. This method decreases the number of comparisons required to reach a decision from $c(c-1)/2$ to $(c-1)$ only, given c classes. For ranking, the distance from boundary (DFB) generated by the SVM was used. DFB was found to have a consistently higher accuracy than other measures. The best performance used both feature sets and achieved an error rate of 8.08%.

MFCC alone were used in [16] for concept classification of consumer videos. The task was to develop a summary in a manner similar to the Bag-of-X for the set of the MFCC vectors extracted for a video. The authors tried three methods: Single Gaussian modeling, Gaussian mixture modeling, and probabilistic latent semantic analysis (pLSA) of a Gaussian component histogram. The pLSA based technique was proposed so that local patterns that represent some concept and that would normally be obscured by the global pattern of the entire soundtrack can be found and used to distinguish between the different concepts. Videos covering 25 concepts were downloaded from YouTube. The dataset was manually filtered to obtain accurate labels, where each video had an average of 3 labels. SVMs were used for classification and the scores were again the distance-to-boundary. All three methods performed relatively well with pLSA providing the best results.

Although the applications were related to concept detection and audio features were used, the goal was to find good distance measures, appropriate classifiers or good parameterization techniques. Optimizing audio feature parameters was generally not the focus, except for a few papers that were trying to improve retrieval accuracy by adding more features that characterize sound.

Some short comings of the prior work included the filtered or controlled datasets and that they mostly did not represent a broad array of concepts. What is more, the prior work did not thoroughly consider the temporal information present in the audio and the benefits of such. Although feature time derivatives were used, they only provide how fast a certain feature changes over time and not the original timing information. Some used statistical methods such as mean and variance which are also not sufficient. Also, for SAI, even though it preserves temporal information, it does not preserve spectral information as well which is important due to the nature of human perception of sound. While the approach of forming a supervector out of concatenation was not examined in the context of video concept detection.

3.2 Audiovisual

Research has been moving into the direction of combining different modalities to improve accuracy. For videos, these are typically the audio and visual modalities.

First, before being able to detect a concept, a classifier needs to learn video and label associations. A classifier can only classify those concepts for which it has prior labels. In order to have more flexibility, it is desirable that a classifier learns concepts in a dataset on its own. This was the concern of [3]. The goal was to automatically annotate consumer videos using an unrestricted amount of labels. A huge dataset from YouTube was utilized and the initial labels used to learn the classifier were the labels and tags provided by the uploaders in their diversity and multilingualism. An algorithm was devised to allow the classifier to use those tags to discover and learn a pool of concepts which would then be used to annotate a new video. The algorithm also allowed for improperly tagged videos to be corrected. Two visual features: Motion Rigidity and CONGAS-Features, and one audio feature: SAI [21], were used. Each feature was used separately or in combination with the others. It was observed that the discovered concepts rely on the native feature space used. For example, audio oriented concepts like accordion were discovered when using the audio feature.

Unlike the previous paper, in all the following papers a predetermined set of labels was used and the focus was on the actual feature extraction process and using audio and visual features and their combination.

In [25], the task was to classify movies into genres according to the affective information. Therefore, the features used were very task specific and came from a cinematographic and psychological perspective. The dataset consisted of manually segmented and labeled movie scenes. Audio and visual features were extracted separately. The visual features included shot duration and lighting key. While the audio features included MFCC, its first derivative, energy, and ZCR. The mean and variance for the feature vectors were computed. Then, the audio and visual feature vectors were concatenated and fed to an SVM classifier. Finally, a sigmoid was fitted to the obtained decision values to get probabilities. It was found that using audio and visual features jointly greatly surpasses using only one modality at a time. Additionally, for this task of detecting affective information, audio features outperformed visual ones.

Dense SIFT was used in [6] as well as audio features (MFCC) for concept detection. The dataset consisted of consumer videos over 25 concepts. Context-based detection where relations between concepts are learned was proposed. The assumption is that concepts co-occur and that utilizing information from different concepts would hence improve detection of other concepts. Therefore, different forms of context fusion were examined including Audio Visual Joint Boosting (AVJB). AVJB fuses SVM kernels generated for the different concepts from the different modalities to generate the optimal kernel for concept detection. The kernel pool, from which AVJB chooses, included several visual kernels as well as one audio kernel. Although, results for AVJB were not improved compared to the other methods explored (described later), it was observed that the most used kernel from the kernel pool was the audio kernel, which goes to show the importance of audio features.

While the prior work only extracted audio and visual features disjointly, the following tried to extract one feature that combines both audio and visual. Also, for the task of concept classification, audio and visual features are extracted jointly in [11]. The dataset consisted of a large number of consumer videos over 21 semantic concepts. An algorithm was devised to extract Short-Term Region Tracks from which visual and audio features were extracted. Visual features were global ones and not local like SIFT because of the diverse content present in the dataset where objects or scenes were seldom repeated. Matching Pursuit (MP) was used as the audio feature. MP decomposes the audio signal into a set of basis functions, and here it was applied to the time window corresponding to the visual region track. It is argued that while MFCC capture the global properties of energy in the signal, MP only focuses on a certain time-frequency partition which keeps it invariant to energy sources other than the object identified. Both audio and visual vectors are then concatenated to generate a short-term audiovisual atom (SAVA). Results indicate that using audiovisual features significantly improves accuracy compared to using individual modalities.

All the previous papers have combined the audio visual modalities in feature space whether synchronously, one feature vector was extracted to represent both audio and visual modalities; or asynchronously, the feature vectors for each modality were extracted independently. The following papers, however, fuse the modalities in semantic space where scores from the different modalities are combined to get one final decision score.

SIFT, STIP (Spatial-temporal interest points), and MFCC were used in [14] for Multimedia Event Detection. All features were represented using the Bag-of-X approach. And SVMs using a chi square kernel were used. The baseline system combined scores from all modalities by averaging to get one final score. This achieved a MAP of 0.633. In addition, the contextual relations among events and concepts were examined. First, classifiers were built for different concepts using either SIFT, STIP or MFCC separately. Then, a graph was constructed to represent the relations between them and events. These relations were determined by the ground truth labels of the training set. After that, the graph was used to improve the detection scores. The results only slightly improved in comparison with the baseline. However, this could be attributed to using only one modality at a time to detect context as opposed to the multimodal approach in the baseline. Additionally, MFCC represented by bags-of-audio-words were found to be adequate for event detection by themselves and along with visual features.

Late fusion was also examined in [6] using global visual features (Gabor texture, Grid Color Moment, Edge Direction Histogram) and MFCC. Weighted averaging between the audio and visual scores was used, where the weights for each concept were determined on a separate validation set. Moreover, late fusion was also applied to context-based concept detection using Audio-Visual Boosted Conditional Random Field (AVBCRF). AVBCRF consists of two steps. First, probabilities of concepts are calculated. Then, the score for each concept is modified by taking into account the detection confidence of the other concepts. Results indicated that exploiting concept relations significantly improves performance over direct late fusion. And once more, audio features were found helpful in a number of ways. For example, using audio eliminated irrelevant videos that visually appeared similar but were different acoustically. It was also found that context fusion using the visual models only did not improve results; they were only improved when adding the audio.

In [27], two methods were proposed for late fusion: Gradient-descent-optimization linear fusion (GLF) and Super-kernel nonlinear fusion (NLF). Both methods start by learning separate classifiers for each modality. Then, GLF learns the optimal weights for combining two modalities using their kernel matrices; GLF can thus be viewed as fusing modalities in feature space. Whereas for NLF, a new kernel matrix is formed using the scores generated from each single modality kernel. A number of features were used to test the methods, those included Color histogram and Speech. Both methods improved performance when tested on the TRECVid 2003 benchmark with NLF performing better than GLF.

The issue of synchronization between the different modalities was not fully examined in the prior literature. Whether cues from the different modalities that are salient for identifying a concept occur at the same time or in the same time period was not explored. Although [6] and [14] exploited the cooccurrence of concepts and examined the extent of contribution of each modality, the features for each modality were independently extracted and the cooccurrence of cues from these modalities was not investigated. The only work that pursued some synchronization was [11] by using SAVA.

Chapter 4

Approach

The first section discusses audio features and their utilization separately for the purpose of concept detection. While the second section explores the different methods of combining audio and visual features for audiovisual concept detection as well as the relations between the two modalities.

4.1 Audio Concept Detection

4.1.1 Digital Audio Analysis

Audio is a continuous-time continuous-valued type of signal. In order to store it and analyze it on a computer, it needs to be digitized. The process starts with sampling the signal (acquiring the signal's values at certain times) using a certain sampling rate adhering to the Nyquist theorem which states that the sampling frequency should be at least twice the highest frequency in the signal. Then the signal is quantized where the amplitudes are also set to certain levels. At this point, the signal is a digital one: discrete-time discrete-valued. Other parameters in a digital audio signal include the sample depth which is the number of bits used to store the value of the sample. Also, the number of channels in an audio recording is the number of streams present– the number of separate signals. A mono recording has only one channel.

As mentioned, the digital audio signal is just a sequence of numbers over time. This is the time domain representation. But sometimes it is more appropriate to represent the signal in terms of its constituent frequencies, and so it is transformed into the frequency domain using the Fourier transform. In the frequency domain, the sequence of numbers represent the amplitudes and the phases (the shift relative to a starting point) of those frequencies instead of time. But audio analysis tools such as the Fourier transform need

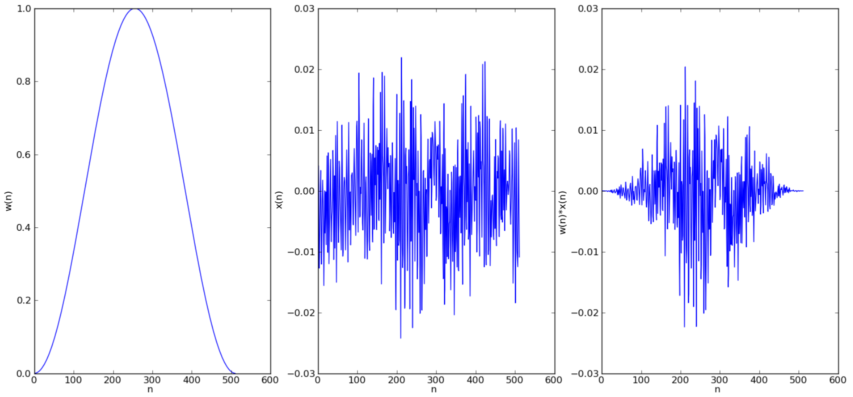


Figure 4.1: Plot of Hann window (left), a sample frame (middle), and the result of applying the window to the frame (right).

a stationary signal. Stationary means that the statistical properties do not vary with time. And due to the nonstationarity of the audio signal, it is divided into short frames (*framing*) where the signal is almost stationary, called quasi-stationary. The size of the frame in milliseconds (ms) depends on the sample rate (to achieve stationarity) where higher sample rates can accommodate larger frames. In addition, the frame size should be short enough to guarantee the stationarity condition but at the same time long enough to have enough information for analysis [24].

This framing is done by multiplying the audio signal by a sliding window function where the result is the signal's values at these positions (inside the window) and the rest is zero. Although a rectangular window function (a constant value) seems like the obvious choice, it might result in discontinuities at the edges which translates poorly when the signal is transformed to the frequency domain. Therefore, other types of windows exist that have a smooth decay at the boundaries and hence give better representations in the frequency domain. A popular window is the Hann window defined as:

$$w(n) = 0.5(1 - (\cos \frac{2\pi n}{N-1})) \tag{4.1}$$

where N is the length (in samples) of the window. Figure 4.1 shows a Hann window and the result of applying it to a sample frame.

Usually successive frames are overlapped in order to get an accurate representation of the original signal, since applying a window slightly alters the original signal. The step size between consecutive frames varies, usually between 33-50% of the frame size [20, 16].

4.1.2 Audio Features

Figure 4.2 shows the audio feature extraction pipeline. Starting with the raw audio signal, the desired features are computed per frame. This results in a set of feature vectors, the size of which depends on the length of the audio signal. Finally, the set is quantized and transformed into the bag-of-audio-words representation.

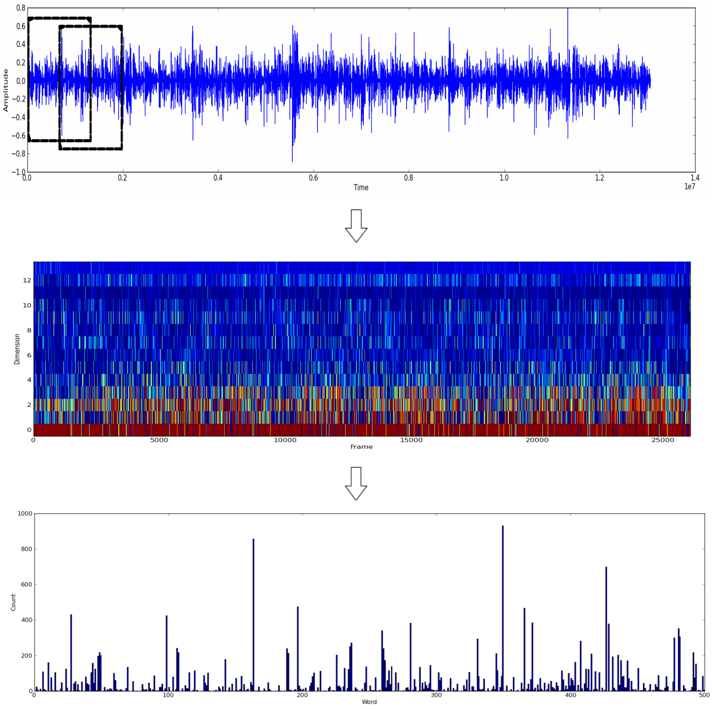


Figure 4.2: This figure shows the audio feature extraction pipeline. First, the audio signal is framed. Then, a feature vector is computed per frame resulting in a sequence of feature vectors. Finally, the feature vectors are transformed into the bag-of-audio-words representation.

The features used in this thesis are MFCC, Energy and high level audio words. Details about MFCC and Energy follow, whereas the higher level audio words are described in the next section.

MFCC

A very common audio feature are the Mel Frequency Cepstral Coefficients (MFCC). Obtaining the MFCC for an audio frame is done through the following steps:

1. The frame is multiplied by a window function such as the Hann window.

2. The windowed frame is transformed to the frequency domain using the Discrete Fourier Transform (DFT). The DFT of a sequence x of length N is defined as:
$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}kn} \tag{4.2}$$

3. The Mel scale, shown in Figure 4.3, is then used to warp the frequencies to reflect the humans' perception of sound. This is done by dividing the frequencies into bins by means of a filter bank equally spaced along the Mel scale as illustrated in Figure 4.4. Then, a weighted average of the log magnitude of the DFT coefficients per bin is computed:
$$Y(i) = \sum_{k} \log |X(k)| H_i(k) \qquad i = 0, \ldots, M-1 \tag{4.3}$$
where M is the number of filters, $H_i(k)$ defines the corresponding weight, and the summation over k is restricted to the DFT coefficients that lie in the corresponding bin. This filtering effectively reduces the dimension of the feature vector to the number of filters used.

4. The inverse DFT (IDFT) is applied to decorrelate and compact most of the energy in very few coefficients. The IDFT is calculated as:
$$c(n) = \frac{1}{M} \sum_{k=0}^{M-1} Y(k) e^{j\frac{2\pi}{M}kn} \qquad n = 0, \ldots, L-1 \tag{4.4}$$
where $Y(k)$ is the output of the kth filter, L is the number of coefficients to be calculated ($L \leq M$), and $c(n)$ is the MFCC vector for each frame. The Discrete Cosine Transform (DCT) is typically used in place of the IDFT because the DFT coefficients of the log magnitude of a signal are real and symmetrical and therefore the DCT is very efficient at computing the IDFT by exploiting these symmetries [24]. The DCT is defined as:
$$c(n) = \sum_{k=0}^{M-1} Y(k) \cos\left[\frac{\pi}{M}\left(k + \frac{1}{2}\right) n\right] \qquad n = 0, \ldots, L-1. \tag{4.5}$$

MFCC represent frequencies that are perceptually relevant by taking several perceptual properties into consideration. The first one is discarding the

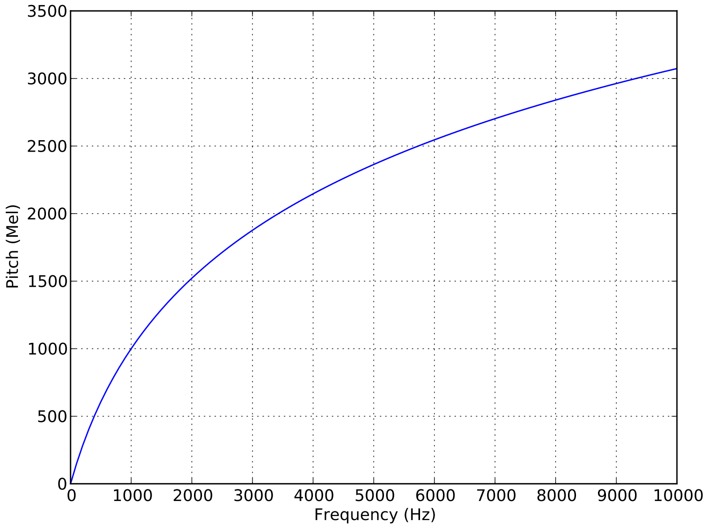


Figure 4.3: The plot shows a mapping between the actual frequencies in Hertz and the perceived pitch in Mel. The scale is approximately linear below 1000 Hz and logarithmic afterwards.

phase information while retaining the magnitude of the spectrum because amplitudes are more important for discrimination. The second lies in the logarithm step because the human ear perceives loudness on a logarithmic basis. And finally there is the Mel step, where the frequency bins over which the frequencies are averaged are not equally spaced but are spaced according to the Mel scale which has a higher resolution at lower frequencies (closely-spaced bins) than at higher ones because the ear is more sensitive to the lower frequencies. The result is the perceptual log magnitude equivalent of the physically measured spectrum [24]. Some works [16, 12] switch the order of the log and the mel step.

The number of coefficients used vary between 5 and 40. The argument is that more coefficients, meaning a higher frequency resolution, help in general audio classification as opposed to just speech (for which MFCC were developed) which does not contain high frequencies. Also the zeroth coefficient which is the mean value of the signal is typically discarded.

Energy

Another descriptor is the energy of the signal frame. The energy in signal processing is the sum of the squares of the amplitudes. The result is related

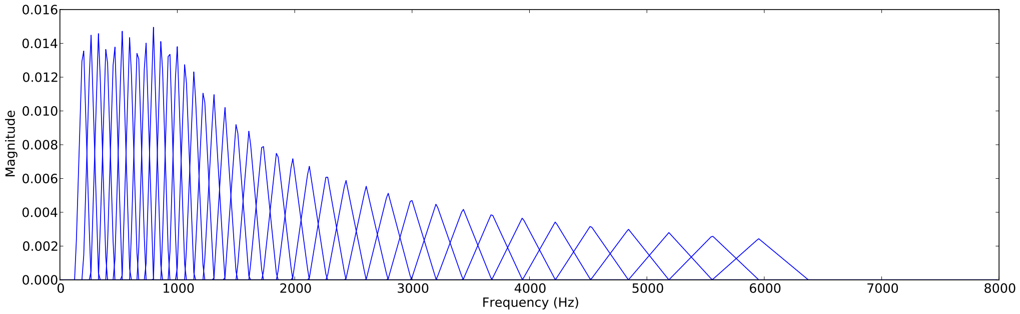


Figure 4.4: The triangular filters are equally spaced along the Mel scale where the lower frequencies have a higher resolution. The triangular shape represents the weights applied to each frequency in its bin. Other implementations apply a constant amplitude to all bins and/or no overlap between consecutive bins.

to the physical energy of the signal. A variant is the normalized energy which allows a comparison between different signals despite varying loudness for instance. The energy in this thesis was computed as follows:

$$energy = \sqrt{\frac{\sum_{i=0}^{N-1} {x_i}^2}{N}} \tag{4.6}$$

4.1.3 Higher Level Audio Words

The bag-of-audio-words representation of an audio signal completely discards the temporal order of the audio words. Therefore, higher level audio words are proposed in order to preserve some temporal information. They are created by concatenating a sequence of audio feature vectors into one supervector. Despite also using the Bag-of-X representation, the temporal information will be implicitly present in the words.

Observing temporal order is hypothesized to be able to capture particular patterns recurring in the same video or in different videos that might be be a signature of a specific concept and therefore aid in identifying it. This is different from just preserving temporal order by using a larger frame size in the extraction of the audio features because a) the patterns will not be captured since the feature extraction has a global view of the audio frame and only captures the overall shape and will miss any inner patterns and b) increasing the frame size most likely violates the stationarity condition in the first place and therefore the results would not be meaningful.

This is also different from differentiating audio features between successive frames and observing how they change along time because the derivatives would show the evolution of the features along an entire video and this sequence would be specific to that video and will not identify shorter patterns and sequences that occur throughout different videos.

4.2 Audiovisual Concept Detection

4.2.1 Audiovisual Fusion

Audio and visual features of a particular concept are usually related. For instance, a video of an interview contains speech. If both visual and audio features were used together, the outcome of a classifier is excepted to improve. Fusion is simply combining multiple modalities together to enhance the prediction accuracy. There are two basic types of fusion.

Early fusion means combining several features together and creating one representation that would be fed to the classifier. This could be as simple as concatenating the feature vectors extracted from the different modalities together into one high dimensional feature vector which is the method used here.

Late Fusion involves training several classifiers–one for each modality–separately, and then the scores from each classifier are combined into one score by weighted averaging.

One advantage of early fusion over late fusion is that only one classifier needs to be trained as opposed to multiple classifiers in late fusion. And one advantage of late fusion is that scores are determined from a semantic point of view which helps exploit one modality's strength, although this could still lead to lower accuracies if one modality is particularly bad for a certain concept.

4.2.2 Localized Audiovisual Fusion

Besides the fact that combining different modalities together might help, the issue of how and when to combine them has not been thoroughly investigated in the literature. The early fusion described in the previous section only combines the independently extracted features of the different modalities into one feature vector and therefore completely ignores any temporal relations that exist between audio and visual features. For example, the fact that a shot of a cat is possibly also accompanied by a meow or a shot of a person is accompanied by speech or song are not discovered. While in reality, it is possible that if only these moments in a video (the ones containing both strong audio and visual cues) are considered for concept detection and the rest either ignored or given lower weights, this could make the system more robust. The question is thus, should audio and visual features be extracted from the same moment in a video?

To gain insight into these shot/audio relations, each video should be segmented, and its segments should then be examined individually on a visual only, audio only, and audiovisual level. Whereas this time for the audiovisual, early fusion is done between a visual keyframe and an audio window extracted from the same point in time. For each visual keyframe, a bag-of-visual-words will be created; and for each audio window, a local bag-of-audio-words will be created (local because normally the bag-of-audio-words represents the entire signal and not just one window). Then, one feature vector representing the window-keyframe pair is formed, namely one bag-of-audiovisual-words.

Since one shot of video lasts for approximately 1/25th of a second, the audio window will need to span a greater length of time around the keyframe because otherwise the audio information will not be enough, contrary to the image which although lasts 1/25th of a second contains 2D information. The length of the audio window is thus a tunable parameter, i.e., how long it should be to represent the event/object appearing in one shot.

In addition to just observing the local scores generated inside the video, one might want to compare these results–looking locally at specific times in the video only both aurally and visually– with the results obtained from the previous early fusion in order to test the robustness described above. Therefore, one final score for each video is needed. This can be calculated by fusing the scores (late fusion) of its window-keyframe pairs. Then the results can be compared.

Chapter 5

Experiments

5.1 Setup

This section presents the general setup of all the experiments including the datasets used as well as the classifier and the evaluation measures.

5.1.1 Datasets

All videos were downloaded from YouTube [2] representing real-world data. Because YouTube primarily hosts (nonprofessional) user generated videos, this dataset is a challenging one owing to the present noise, be it in the actual video shots or in the tags associated with the video where the labels might not correspond to the content of the video. In addition, for audio especially, videos included accompanying background music which do not reflect the expected sounds of the object in the videos (e.g. cats and meowing).

Controlled Dataset

This dataset consists of clips that were manually segmented from different videos to create true labels for three concepts: speech, music and silence. A total of 75 clips were created, divided over the three concepts as follows: Speech 28, Music 28, Silence 19.

Dataset I

This dataset was provided by Multimedia Analysis and Data Mining (MADM) group[1]. The labels used were those provided by the original YouTube uploaders. In total, 1140 videos spanning ten semantic concepts were used.

[1] http://madm.dfki.de/downloads

The dataset contained: airplane flying (100 videos), beach (102 videos), cats (186 videos), dogs (118 videos), emergency vehicle (102 videos), explosion (116 videos), interview (115 videos), race (101 videos), riot (100 videos), and singing (100 videos). These specific concepts were chosen under the assumption that they are more audio oriented. For instance, an emergency vehicle has a very distinct siren as opposed to a concept like flower which does not have any particularly strong audio associations. It should be mentioned that a random check on some videos revealed duplicates that might have slightly affected the results.

Dataset II

Visual features were only available for a subset of the previous set, though all ten concepts were still represented. This dataset contained 1003 videos distributed as follows: airplane flying (99 videos), beach (99 videos), cats (98 videos), dogs (117 videos), emergency vehicle (95 videos), explosion (99 videos), interview (99 videos), race (99 videos), riot (99 videos), and singing (99 videos).

5.1.2 Classification and Evaluation Measures

The training and testing subsets were created by randomly splitting the videos per concept in half. An SVM was used to learn the concepts due to its good generalization ability when only a small training dataset is available. It utilized a chi square kernel which is suitable for dealing with histogram like features [14]. The probabilistic scores were then generated via a sigmoid fit. Finally, the performance measures used were Average Precision (AP) and Mean Average Precision (MAP).

5.2 Audio Concept Detection

5.2.1 Audio Features

The Gstreamer framework [1] was first used to extract the audio stream from the videos and obtain a mono channel with a sample rate of 22050 Hertz. Then, the Yaafe toolbox [5] was used to extract MFCC and energy features. The overlap size between successive frames over which the features are computed is 50%.

MFCC The parameters for MFCC were as follows:

- A Hann window is applied to the frame before FFT is computed.
- The zeroth cepstral coefficient is discarded.
- Only 13 cepstral coefficients were used.

Energy The energy in Yaafe was computed as per Equation 4.6.

5.2.2 Experiment 1 - Controlled Setting

The purpose of this experiment was to verify that the chosen classifier and audio features would correctly identify the different classes. MFCC and Energy were extracted for the Controlled Dataset using two frame sizes: 46 ms and 32 ms. The experiment was conducted twice, once with just the MFCC features and another with the additional energy which was appended to the end of the MFCC vector. Finally, all feature vectors from all the videos were used to build a codebook of 500 words which was used to create the bag-of-audio-words.

Results

APs achieved for the different parameter settings are shown in Figure 5.1. The figure also shows the results for random guessing. These results indicate that the three classes were easily separable and that the performance was stable across the different parameter settings. And more importantly, the MAP of using audio features (0.977 for *MFCC-46ms*) shows a threefold improvement over that of random guessing (0.333).

5.2.3 Experiment 2 - Parameter Tuning

Following the results of the previous experiment, the purpose of this experiment was to verify that audio descriptors can be used in semantic concept detection for video classification. It is therefore similar to *Experiment 1* but applied on Dataset I with the broader range of classes.

Several parameters can be tuned to find the optimal combination for the task of concept detection. The frame size from which the features are computed is one parameter. The other parameter is the number of MFCC to keep; however, in this thesis, a standard 13-coefficient vector was used. And the final parameter is the codebook size which represents the number of words in the bag-of-audio-words. This parameter is the input k to the k-means algorithm.

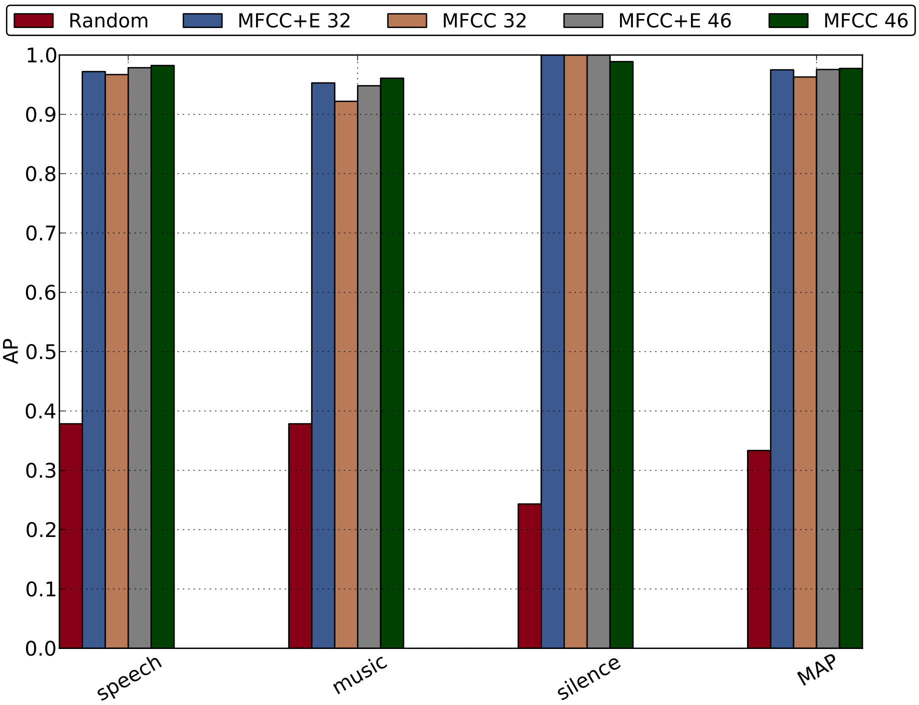


Figure 5.1: Comparison of different frame sizes and the additional usage of the energy of the audio signal over the Controlled Dataset. Performance is stable across all versions.

Experiment 2.1 - Frame Size and Energy

Based on [24], reasonable frame sizes for an audio signal with a sample rate of 22050 Hz would be in the range 23-47 ms. Therefore, three frame sizes: 25 ms, 32 ms, and 46 ms were used. For this experiment, only a subsample of feature vectors were used to build a codebook of 500 audio words.

Results of using the different frame sizes and the additional energy are shown in Figure 5.2. It is observed that generally the smaller the window size, the better the performance. This is contradictory to the results in *Experiment 1* where the larger window size performed better. The greater amount of training samples could be the reason. Regardless, the difference between *MFCC-46ms* and *MFCC+E-32ms* was not significant anyway, a mere 0.02%. Additionally, the system performed better without using the energy feature. Here, *MFCC-25ms* had the highest MAP of 0.440, about 2% greater than the rest and more than a fourfold increase over random guessing. Also, generally results for the concept *interview* (mostly speech dialogues) were high (AP 0.878) which verifies the appropriateness of MFCC for speech.

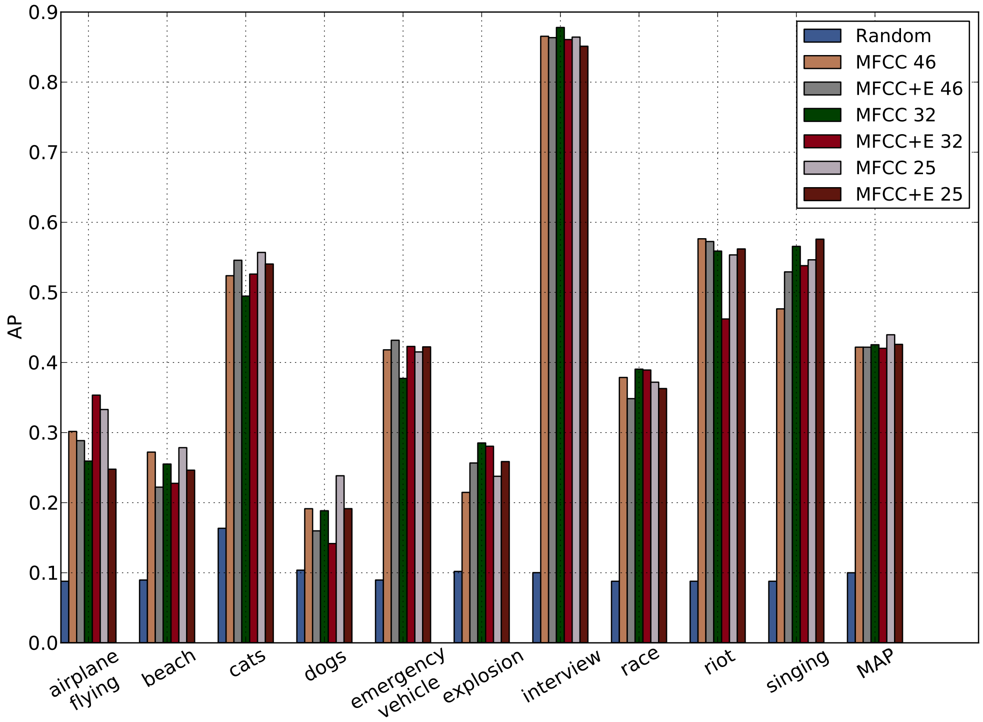


Figure 5.2: Comparison of three frame sizes and the additional usage of the energy of the audio signal over Dataset I. Results for all versions were comparable. The frame size of 25 ms without using the energy (*MFCC-25ms*) performed the best.

Experiment 2.2 - Codebook Sizes

Since finer quantization could improve performance, different codebook sizes were evaluated. Also, a larger subsample of feature vectors was used for clustering to accommodate the larger codebook sizes. For this experiment, the frame size of 25 ms (*MFCC-25ms*) was used due to the previous results.

Ten different codebook sizes were used starting with 500 for comparison with *Experiment 2.1* to check whether using a larger subsample of vectors would provide the k-means algorithm with better clusters.

Figure 5.3 shows the MAPs for the codebook sizes experiment. It is noted that the codebook size 500 gave a worse result (MAP 0.428) than *Experiment 2.1* (MAP 0.440); this could be due to k-means stochastic properties, where better or worse clusters were generated. Although there was no consistent winner throughout the concepts, the MAPs improved with increasing codebook size. But after codebook size 2000 (MAP 0.457), the improvements were not significant with the highest observed at size 4000 (MAP 0.470) which is only a 1.3% increase. For the following experiments, *MFCC-25ms* with a codebook of size 2000 will be used and will be referred to as *MFCC-25ms*.

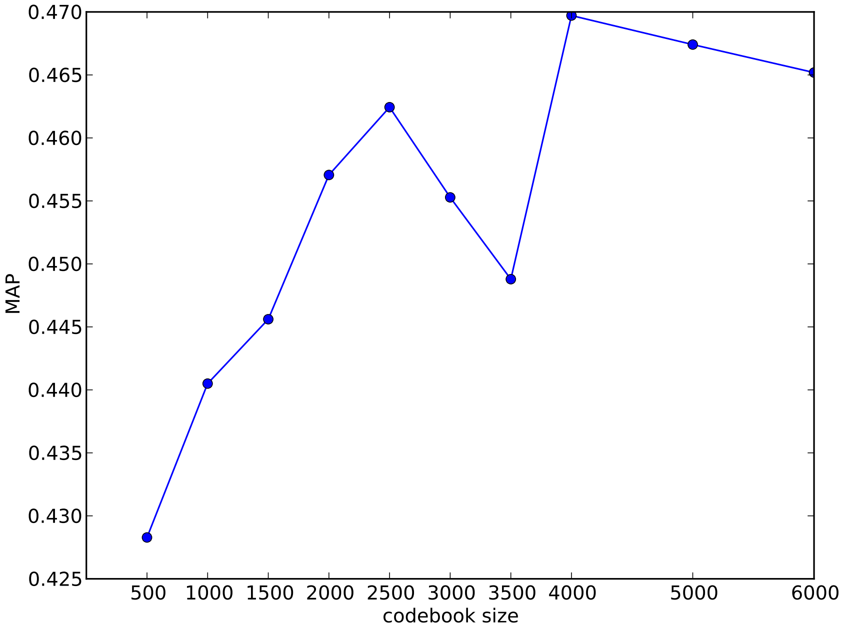


Figure 5.3: MAPs for the different codebook sizes. Improvements after codebook size of 2000 are not significant, the highest increase of 1.3% is observed at size 4000.

5.2.4 Experiment 3 - Higher Level Audio Words

This experiment aims at verifying the importance of timing information and tests the higher level audio words explained in Section 4.1.3 for the task of concept detection.

The higher level audio words were created by concatenating a sequence of 40 MFCC vectors (*MFCC-25ms*), corresponding to 0.5 second of audio, to form a 520-dimensional vector. Then, a codebook of size 2000 was built and used for feature vector quantization.

Results

Results are shown in Figure 5.4 along with *MFCC-25ms* for comparison. It can be observed that the performance of higher level audio words in terms of MAP is slightly increased (0.6% higher) than that of the standard audio words. However, for the concept *interview* which mostly contains speech dialogues, AP increased by 3%, possibly indicating that implicitly the underlying structures of language have been captured. Similarly for *singing* at 9% and the possibly repeating rhythm. There are obviously improvements for other concepts as well; however, it is not obvious what kind of patterns have been captured.

These results prove that preserving temporal structure does in fact help for concept detection. It might be worth investigating whether increasing the codebook size, due to the high dimensionality of the vectors, would improve accuracy. Also, varying the length of the sequence between longer and shorter than half a second to check whether this improves accuracy.

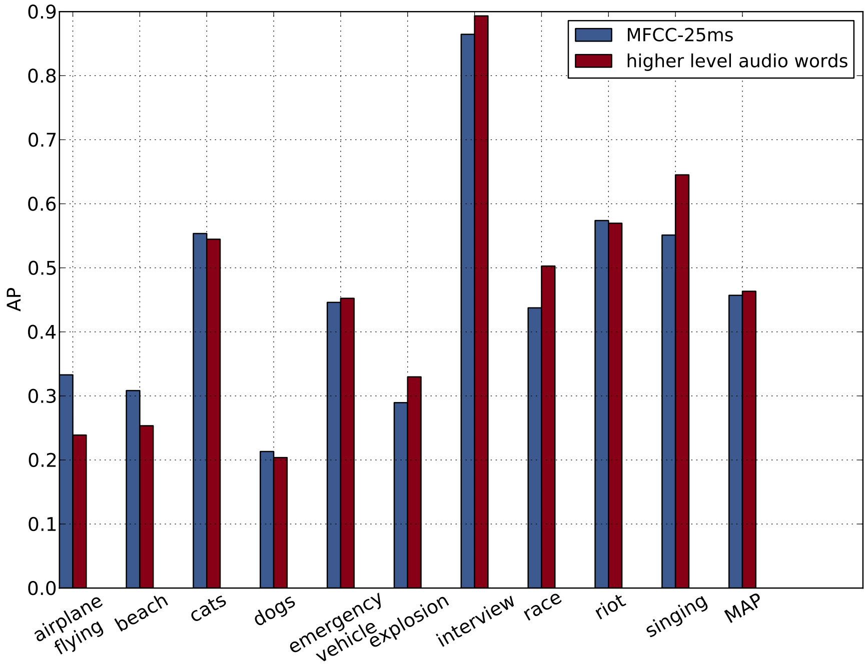


Figure 5.4: Comparison of higher level audio words with standard audio words. The higher level audio words are slightly better in general. Though some individual concepts are much improved.

5.3 Audiovisual Concept Detection

While the previous experiments used only audio features, the following set of experiments combine audio and visual features to improve concept detection. The experiments are performed using Dataset II.

5.3.1 Visual Features

For each video, keyframes were extracted based on change detection. Dense SIFT features are then extracted from each keyframe. And finally bags-of-visual-words from a codebook of size 2000 were created. This forms the visual baseline.

5.3.2 Experiment 4 - Audiovisual Fusion

The purpose of this experiment was to measure improvements added by audio when combined with visual features. The audio baseline used the bag-of-audio-words created from *MFCC-25ms* and a codebook of size 2000. Both early fusion and late fusion were examined.

Early Fusion

For early fusion, all keyframes' bags-of-visual-words belonging to the same video were averaged to get one bag per video. This was then concatenated with the bags-of-audio-words generating a 4000-dimension joint histogram.

Late Fusion

For late fusion, the weight used to combine the separate audio and visual scores was determined using a 3-fold cross validation on the training set. This is a global weight, meaning it was determined by and for all concepts instead of estimating a different weight for each concept. This choice was based on the fact that the dataset is small and hence has few training videos per concept. Results of the three folds and their average are shown in Figure 5.5. The equal weights (0.5) for visual and audio features verify that the chosen concepts are audio oriented.

Results

Figure 5.6 shows the AP attained per concept for each modality separately and for the two fusion methods. Note that for visual features alone, also all keyframe scores were averaged to get the video score. As can be seen, both

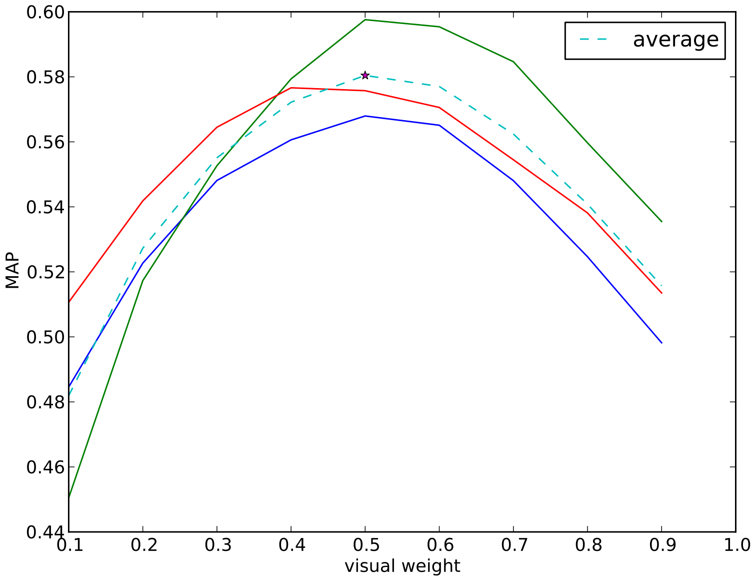


Figure 5.5: Performance in terms of MAP for each visual weight over three folds in cross validation. Also shown is their average. A star marks the best MAP at a weight of 0.5.

forms of fusion improve results significantly than using one single modality. And early fusion is generally better than late fusion, though both are comparable. The MAP for early fusion was 7.4% and 18% better than using the singular visual and audio modalities resepectively.

When early fusion was performed using higher level audio words, the results were worse. In fact, the higher level audio words on their own for Dataset II were worse than the standard audio words (*MFCC-25ms*). Possible explanations for this include the existence of two patterns that might be similar but one is farther stretched than the other (Dynamic Time Warping) and therefore the similarity was not captured by the higher level audio words which are restricted by a certain length (number of consecutive vectors) for the pattern. And of course the duplicates issue for Dataset I might be involved.

5.3.3 Experiment 5 - Localized Audiovisual Fusion

The purpose of this experiment was to examine the existence of correlations between audio and visual cues and to investigate whether certain cues occur together frequently and whether this might improve the classification

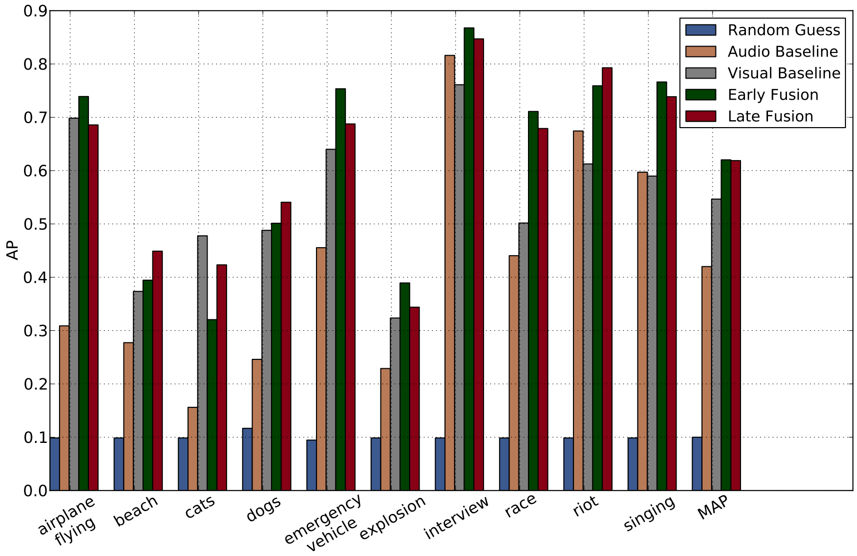


Figure 5.6: Comparison of different modalities and fusing them using early and late fusion. Both fusion methods improve accuracy significantly than using single modalities.

accuracy if taken into consideration.

For the late fusion of all segments (keyframes and/or audio windows), two methods were tried out. The first was just averaging the scores (*Avg*) while the second was average plus maximum (*AvgMax*) which should give rise to the overall score of a video if the concept is strongly present in one shot or audio window.

The experiment consisted of three parts. First, a suitable size for the audio window was found, then late fusion was carried out using the two methods. Second, for the visual features, late fusion over the keyframes' score was carried out using the two methods. And third, the audiovisual fusion was performed using the best parameters determined by parts 1 and 2.

Part 1 - Audio Window Size

To determine an appropriate window size to capture around the keyframe, ten window sizes were tried from 1 second to 10 seconds (equivalent to 80-800 audio words). This range was chosen to later (in audiovisual fusion) accommodate the keyframes extraction, so that the audio windows corresponding to two successive keyframes would not overlap. First, the whole sequence of audio words for each video was segmented into overlapping windows. The entire sequence was used and not only those surrounding a keyframe because this allows a true look at whether using local audio windows would be beneficial without being constrained by the keyframes' locations. Also, the windows were overlapped using different amounts so that the same amount of training data would be available for all window sizes. Then, for each window, a local bag-of-audio-words was created using the same codebook as *MFCC-25ms*; those were fed to the classifier. Finally, the score of a video was calculated by combining the scores from the individual windows using either *Avg* or *AvgMax*.

Figure 5.7 shows the MAP achieved for both methods. Using averaging clearly outperforms the average plus maximum strategy. And the best MAP of 0.383 was achieved by the 8-second window. This result is, however, significantly worse than that achieved by the global bag-of-audio-words in *Experiment 4* at 0.420 which shows that, for audio, the overall distribution of audio words as shown by the global bag-of-audio-words was more discriminant of the different concepts.

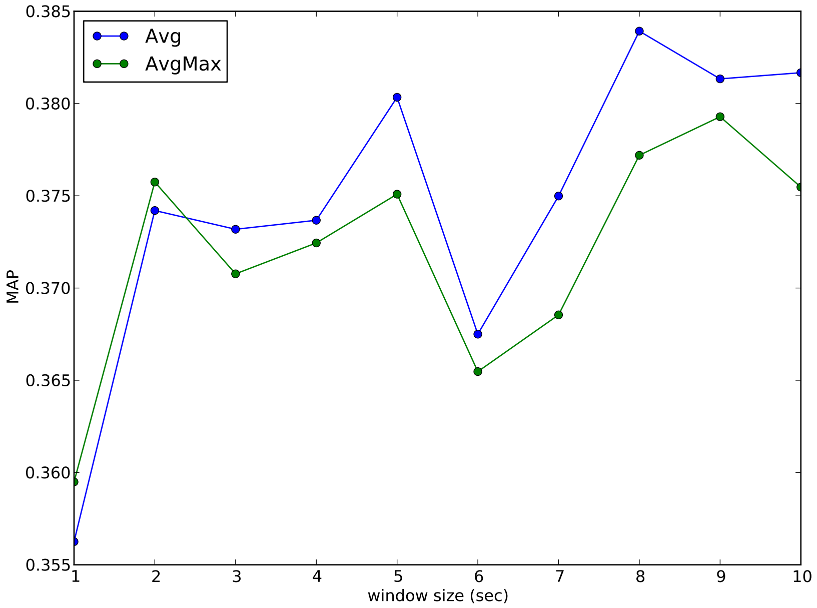


Figure 5.7: Comparison of different audio window sizes for both *Avg* and *AvgMax*. The best MAP of 0.383 was for the 8-second window with *Avg*.

Part 2 - Visual Keyframes

SVMs were trained using the bag-of-visual-words available for each keyframe. For each video, the scores for the keyframes were combined either by *Avg* or *AvgMax*. APs for the two fusion methods are shown in Figure 5.8. Once again *Avg* (MAP 0.547) outperformed *AvgMax* (MAP 0.528).

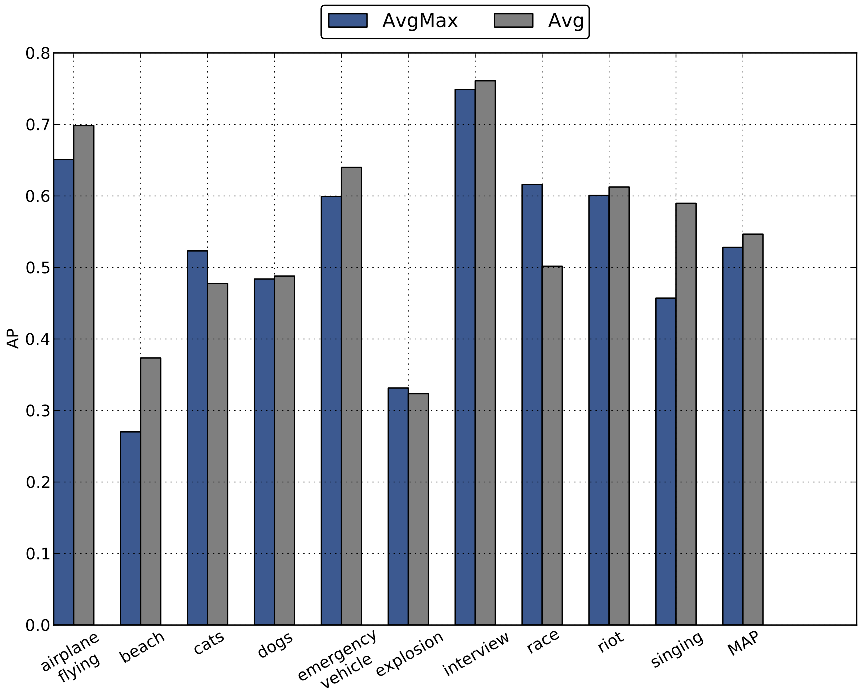
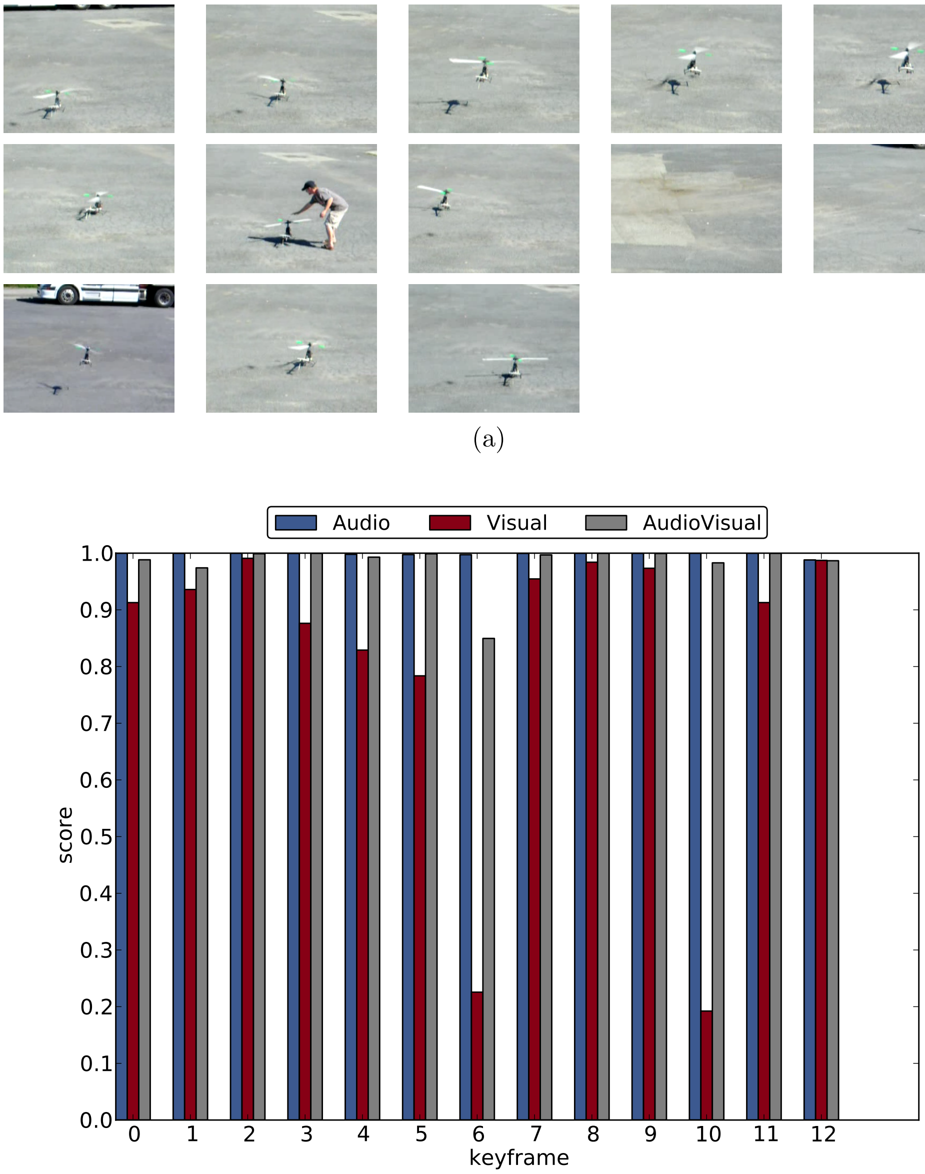


Figure 5.8: APs for using *Avg* and *AvgMax* for the visual modality only. Again *Avg* provided better results.

Part 3 - Audiovisual fusion

For this experiment, the audio words in an 8-second time window were extracted (local bag-of-audio-words) from around the keyframes, i.e., the keyframes were the centers of the audio window. Early fusion was examined by forming a bag-of-audiovisual-words which is a simple concatenation of the bag-of-audio-words of the audio window with the bag-of-visual-words of the keyframe.

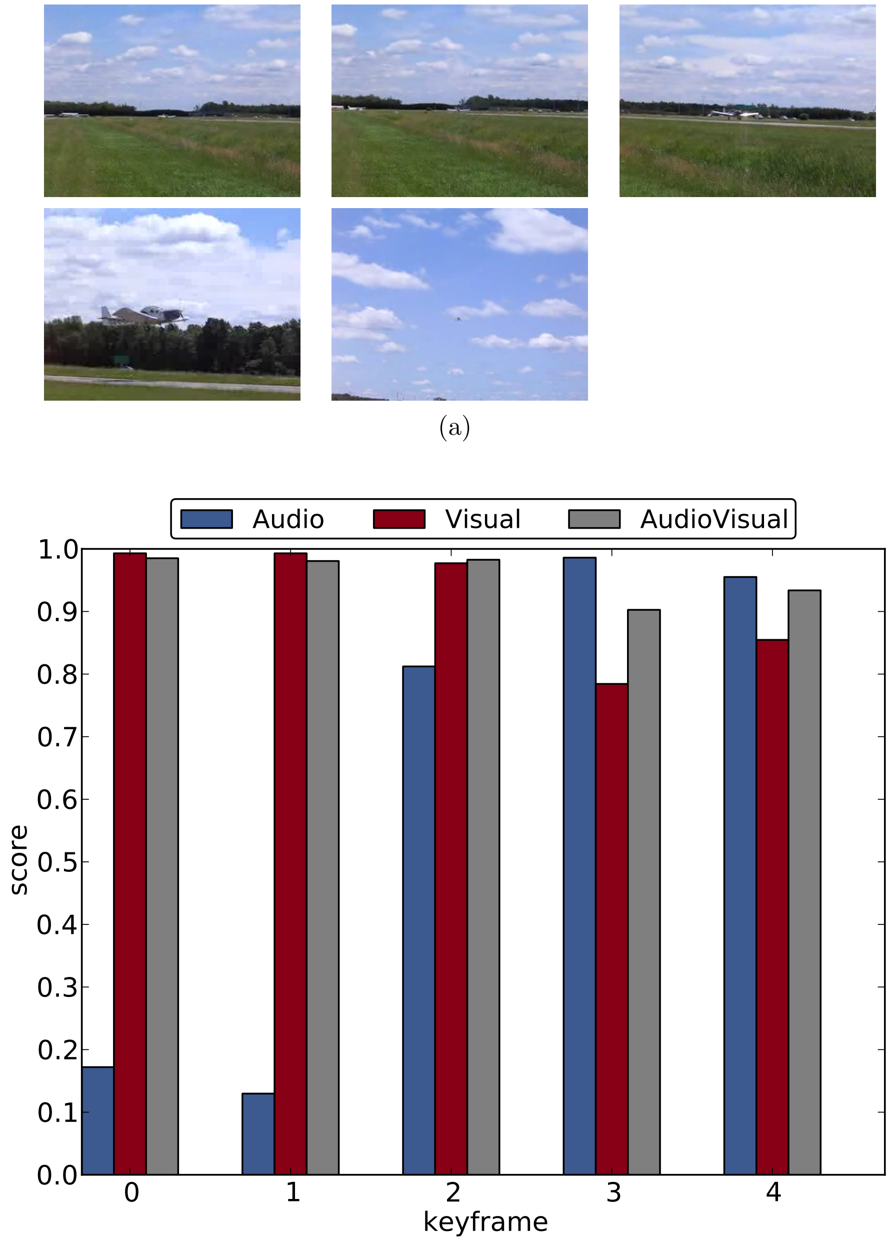
In an attempt to provide some insight into these audio and visual relations, two examples from the concept *airplane flying* are shown in Figures 5.9 and 5.10. Figure 5.9 shows the extracted keyframes as well as the individual scores for each keyframe, audio window, and their fusion for the first example. As seen in the keyframes, the video featured a helicopter and the accompanying audio was the high frequency buzz of the engine. For this video, both audio and visual cues occurring at the same time were strongly indicative of the concept *airplane flying.* On the other hand, for the second example shown in Figure 5.10, the audio, for the first few shots, was just noise (wind) as the airplane was far from the recording microphone but then the airplane got closer and the engine sound was audible. However, by using audiovisual fusion, the scores were still high indicating the presence of the concept *airplane flying.*



(a)

(b)

Figure 5.9: a) These are the keyframes extracted for the first example ordered from left to right and top to bottom. The aircraft is visible and sounding throughout the video. Although the aircraft was out of sight in the eighth frame, it was still heard as evident by the audio score below. b) This plot shows the scores for each keyframe, audio window, and their fusion. Both audio and visual cues were strong throughout. The low visual scores at 6 and 10 resulted from other objects entering the scene but still the audiovisual fusion helped alleviate this by using the strong audio cue.



(a)

(b)

Figure 5.10: a) These are the keyframes extracted for the second example. As can be seen from the first two frames, the aircraft was far from the recording microphone and hence the low audio scores. b) This plot shows the scores for each keyframe, audio window, and their fusion. Audio and visual cues were only strong together for a part of the video. But again the fusion helped neutralize these effects and provide a strong score.

Finally, to observe whether using audiovisual moments only would improve accuracy, the scores obtained from the SVMs for the audiovisual bags were averaged (*Avg*) to get one score per video. The APs for the localized audiovisual fusion as well as the global early fusion of *Experiment 4* are compared in Figure 5.11. The MAP for audiovisual fusion (0.546) was a little worse than visual alone (0.547) and worse than the early fusion in the previous experiment (0.62).

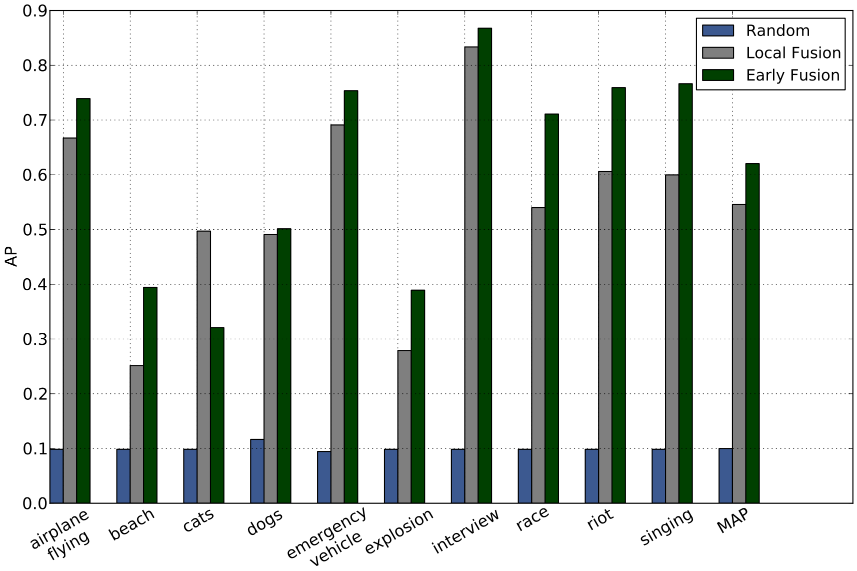


Figure 5.11: APs for localized audiovisual fusion using *Avg*. A staggering 18% increase for the concept cat.

Due to the keyframes extraction algorithm, each keyframe is very likely to contain the object/event of its concept. But when extracting audio features from the location of the keyframes, it seems that these audio cues (if present) most of the time do not correspond to the respective label. In other words, strong audio cues occur elsewhere in the video and not at the same time as the visual cues, at least for this dataset. In general, the object producing a sound might not be present in a shot or the object could be in view but far from the recording microphone or only starts producing sounds later in the video. One final note, it is not clear why *Avg* would perform better than *AvgMax*. The small dataset could be the reason.

Chapter 6

Conclusion

The thesis investigated audio features for automatic video tagging. MFCC, energy as well as high level audio words were explored for the task of concept detection. Experiments were conducted using only audio features and using both audio and visual features in early and late fusion setups. When using audio features alone, results have shown that MFCC represented by the bag-of-audio-words have a great capability in discriminating between a broad range of concepts consistent with findings in [14]. And the higher level audio words were shown to slightly enhance the results overall and significantly improve it for certain concepts.

For further improvements, it might be worth investigating the use of more coefficients for MFCC for this general concept detection. And regarding the higher level audio words, parameters such as the number of consecutive MFCC vectors that form one higher level audio word as well as the codebook size (given the high dimensionality) need to be examined.

Although in the literature, additional audio features beside MFCC are used, these require some knowledge of the types of audio present in the concepts to be able to find robust descriptors. However, for general concept detection with the diverse array of concepts, one can suffice with features representative of the spectral (MFCC) or temporal (higher level audio words or SAI) information that mimic the natural human perception of sound.

Moreover, audiovisual fusion was carried out using early fusion and late fusion. Both fusion methods significantly improved accuracy over using single modalities. And early fusion was generally better, though results of both were comparable.

Further more, early fusion was performed locally on keyframe and audio window pairs to examine whether visual and audio cues salient for detecting a concept occur concurrently and to test whether system robustness increases by utilizing only those moments of cooccurrence. It was observed that this does not always hold as strong audio cues sometimes occur at another point in time.

For audiovisual fusion, one short coming was using already available keyframes and extracting audio windows at those positions which makes the visual modality the dominant one. Also, the audio window was centered around the keyframe, where in fact the keyframe might be at the end of an event or the start of one and the audio cue could be missed. A fairer method would be to regularly sample both visual keyframes and audio windows. Also different synchronization schemes should be looked into, since the case might remain that audio and visual cues do not occur at the exact same moment but could occur in the same time period or in sequence. Further suggestions include exploring other visual features. And generally using a larger dataset with more training videos would be beneficial.

Bibliography

[1] Gstreamer Framework. http://gstreamer.freedesktop.org/.

[2] Youtube - Broadcast Yourself. http://www.youtube.com/.

[3] Hrishikesh Aradhye, George Toderici, and Jay Yagnik. Video2Text: Learning to Annotate Video Content. *ICDM Workshop on Internet Multimedia Mining*, 2009.

[4] Pradeep K. Atrey, M. Anwar Hossain, Abdulmotaleb El Saddik, and Mohan S. Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Springer Multimedia Systems*, 16(6):345 –379, 2010.

[5] B.Mathieu, S.Essid, T.Fillon, J.Prado, and G.Richard. Yaafe, an Easy to Use and Efficient Audio Feature Extraction Software. In *11th ISMIR conference, Utrecht, Netherlands*, 2010.

[6] Shih-Fu Chang, Dan Ellis, Wei Jiang, Keansub Lee, Akira Yanagawa, Alexander C. Loui, and Jiebo Luo. Large-Scale Multimodal Semantic Concept Detection for Consumer Video. In *ACM MIR*, 2007.

[7] C.V. Cotton and D.P.W. Ellis. Audio Fingerprinting to Identify Multiple Videos of an Event. In *2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 2386 –2389, March 2010.

[8] Helenca Duxans, Xavier Anguera, and David Conejero. Audio-Based Soccer Game Summarization, 2009.

[9] A.J. Eronen, V.T. Peltonen, J.T. Tuomi, A.P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi. Audio-Based Context Recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):321 – 329, January 2006.

[10] Jonathan T. Foote. Content-Based Retrieval of Music and Audio. In *Storage and Retrieval for Image and Video Databases*, 1997.

[11] Harald Gebhard and Lars Lindner. Short-Term Audio-Visual Atoms for Generic Video Concept Classification. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 6:5–14, 2010.

[12] Guodong Guo and S.Z. Li. Content-Based Audio Classification and Retrieval by Support Vector Machines. *IEEE Transactions on Neural Networks*, 14(1):209 – 215, January 2003.

[13] H. Hermansky. Mel cepstrum, deltas, double-deltas, .. - what else is new? Robust Methods for Speech Recognition in Adverse Conditions, Tampere, Finland, 1999.

[14] Yu-Gang Jiang and Dan Ellis. Columbia-UCF TRECVID2010 Multimedia Event Detection: Combining Multiple Modalities, Contextual Concepts, and Temporal Matching, 2010.

[15] M.N. Kaynak, Qi Zhi, A.D. Cheok, K. Sengupta, Zhang Jian, and Ko Chi Chung. Analysis of Lip Geometric Features for Audio-Visual Speech Recognition. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(4):564 – 570, July 2004.

[16] K. Lee and D.P.W. Ellis. Audio-Based Semantic Concept Classification for Consumer Video. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1406 –1416, August 2010.

[17] Beth Logan. Mel frequency cepstral coefficients for music modeling. In *In International Symposium on Music Information Retrieval*, 2000.

[18] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints, 2003.

[19] Krystian Mikolajczyk and Tinne Tuytelaars. Local Image Features. In *Encyclopedia of Biometrics*, pages 1–5. Springer, 2008.

[20] L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Pearson Education Signal Processing Series. Pearson Education, 1993.

[21] Martin Rehn, Richard F. Lyon, Samy Bengio, Thomas C. Walters, and Gal Chechik. Sound Ranking Using Auditory Sparse-Code Representations. In *ICML 2009 Workshop Sparse Methods for Music Audio*, 2009.

[22] Sigurdur Sigurdsson, Kaare Brandt Petersen, and Tue Lehn-Schiøler. Mel Frequency Cepstral Coefficients: An Evaluation of Robustness of MP3 Encoded Music. In *Proceedings of the 7th International Conference on Music Information Retrieval*, pages 286–289, October 2006.

[23] Alan F. Smeaton, Paul Over, and Wessel Kraaij. Evaluation campaigns and TRECVid. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, New York, NY, USA, 2006. ACM Press.

[24] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition.* Elsevier Inc, 4th edition, 2009.

[25] Hee Lin Wang and Loong-Fah Cheong. Affective Understanding in Film. *IEEE Transactions on Circuits and Systems for Video Technology*, 16:14–20, 2006.

[26] E. Wold, T. Blum, D. Keislar, and J. Wheaten. Content-Based Classification, Search, and Retrieval of Audio. *IEEE Multimedia*, 3(3):27 –36, Fall 1996.

[27] Y. Wu, C.-Y. Lin, E.Y. Chang, and J.R. Smith. Multimodal Information Fusion for Video Concept Detection. In *2004 International Conference on Image Processing (ICIP 2004)*, volume 4, pages 2391 – 2394, October 2004.

[28] Dina Younes and Pavel Steffan. Utilizing MFCC for Voice Intensity Determination. *Annual Journal of Electronics*, 4(2):162 – 164, 2010.