Automatic, Context-of-capture-based Categorization, Structure Detection and Segmentation of News Telecasts

Arne Jacobs, George T. Ioannidis (Task leader)
Center for Computing Technologies (TZI), University of Bremen, Germany
{jarne, george.ioannidis}@tzi.de
Stavros Christodoulakis, Nektarios Moumoutzis, Stratos Georgoulakis, Yiannis Papachristoudis
Laboratory of Distributed Multimedia Information Systems and Applications
Technical University of Crete (TUC/MUSIC), Chania, Greece
{stavros, nektar, stratos, ipapachr}@ced.tuc.gr

Abstract

News programs are used for instant and comprehensive reporting of what is happening around the world. Automatic story segmentation and indexing techniques provide a convenient way to store, browse and retrieve news stories based on the user preferences. The objective of the work reported here is to provide an automatic, context-of-capture categorization, structure detection and segmentation of news broadcasts employing a multimodal semantic based approach. We assume that news broadcasts can be described with context-free grammars that specify their structural characteristics. For parsing of a news video following a corresponding grammar, we propose a system consisting of two main types of interoperating units: The recognizer unit consisting of several modules and a parser unit. The recognizer modules (audio, video and semantic recognizer) analyze the telecast and each one identifies hypothesized instances of features in the audiovisual input. Such features could range from upper-level concepts like a specific person appearing in a shot (e.g., the anchor), the appearance of a certain series of frames (e.g., the introduction sequence, which many news broadcasts have in their beginning), the possible topic in a news story, to low-level, e.g., the similarity of two frames. A probabilistic parser using a probabilistic context-free grammar analyzes the identifications provided by the recognizers. In essence, the recognizers provide the parser with actual lexical tokens as a lexical analyzer would provide to a programming language parser. The grammar represents the possible structures the news telecast may have (the grammar is different for each type of news telecast), so the parser can identify the exact structure of this telecast.

Categories and Subject Descriptors

H.3.1 Content Analysis and Indexing; H.3.7 Digital Libraries; I.5.4 Pattern Recognition: Applications; F.4.2 Grammars and Other Rewriting Systems

General Terms

Algorithms, Design

Keywords

News, Segmentation, Classification, Audio Analysis, Video Analysis, Semantic Analysis, Probabilistic Grammars

1 Introduction

News programs are a valuable source of information regarding what is happening in the world. One of the definitions by the U.S. National Institute of Standards and Technologies (NIST) for the news stories is that a news story is a segment of a news broadcast with a coherent news focus, which may include political issues, finance reporting, weather forecast, sports reporting, etc. (http://www-nlpir.nist.gov/projects/tv2004/tv2004.html#2.2). Other parts inside a news telecast cover commercials, previews, introduction, outroduction etc. The coverage of a news program is very comprehensive, and it is likely that individual viewers are interested in only a few stories out of the complete news broadcast. Automatic story segmentation and indexing techniques provide a convenient way to store, browse and retrieve news stories based on the user preferences.

News segmentation in broadcast videos is an emerging problem, and many researchers from various areas, such as multimedia, information retrieval and video processing, are interested in it. To address this problem, low-level features (referring to the audio and/or the visual signal of the video) could be extracted and then higher level features (such as the identification of news story boundaries and their classification) can be inferred. The problem of relating low-level features with higher-level ones that correspond to the human-like understanding of video content is the well-known problem of bridging the 'semantic gap' and a solution to it is necessary for effective retrieval performance. The problem is very difficult in general. In the Delos NoE, specifically in task 3.9, "Automatic, context-of-capture based Categorization, Structure Detection, and Segmentation of News Telecasts", our approach to bridge this semantic gap is twofold: Firstly by restricting the application domain to news videos, and secondly by exploiting the combination of multimodal analysis with semantic analysis.

A first key observation to help bridge the semantic gap in the news video domain is that semantic concepts in news videos are conventionalized in many ways and this fact can be exploited. One convention is that segments in news telecasts do not appear in arbitrary order. Instead, most news telecasts follow a relatively strict scheme that determines the order of segments. E.g., different news stories are usually separated by anchor shots containing the presentation of the story that follows. This telecast structure allows the viewer to easily recognize different segments. Each news program has its own structural model.

We assume that these news format models can be described with context-free grammars. Such a grammar can effectively support a multimodal segmentation process by removing ambiguities in classification, or by associating certain audiovisual cues with segment classes (e.g., news story, presentation). It models all interesting constraints in visual, audio and semantic features that are partly due to the way news programs are produced and partly to the habits and ways of work of the journalists and other agents related to the news production. Combining these features of different modalities through an appropriate grammar we achieve a multimodal analysis of the news telecasts.

For parsing of a news video following a corresponding model, we propose a system consisting of two main types of interoperating units: The recognizer unit consisting of several modules and a parser unit. The recognizer modules analyze the telecast and each one identifies hypothesized instances of features in the audiovisual input. Such features could range from upper-level concepts like a specific person appearing in a shot (e.g., the anchor), the appearance of a certain series of frames (e.g., the introduction sequence, which many news broadcasts have in their beginning), to low-level, e.g., the similarity of two frames. A probabilistic parser using a probabilistic grammar analyzes the identifications provided by the recognizers. In essence, the recognizers provide the parser with actual lexical tokens such as a lexical analyzer would provide to a programming language parser. The grammar represents the possible structures the news telecast may have, so the parser can identify the exact structure of this telecast.

In the rest of this paper we first present related work in the area of news segmentation and classification and we justify the suitability of our grammar-based approach (section 2). Section 3 presents the overall design adopted. Section 4 presents the components of the architecture. Section 5 discusses in more detail the parsing process and how the parser communicates with the recognizer modules. Section 5 concludes.

2 Related work and suitability of our approach

The typical way of integrating multimodal features in order to achieve News Segmentation and Classification (i.e. extracting and classifying news stories from news videos) is through statistical methods such as Hidden Markov Models (Dimitrova, Agnihotri and Wei 2000, Brand and Kettnaker 2000, Boreczky and Wilcox 1998, Greiff *et alii* 2001, Eickeler and Muller 1999, Huang *et alii* 1999). In these approaches, the basic hypothesis is that the multimodal (low-level) features observed can be considered Markovian in some feature space and that efficient training data exists to automatically learn a suitable model to characterize data.

However, in other similar domains such as recognition of visual activities and interactions in videos (e.g. surveillance videos, gesture videos, etc.) another complementary approach based on various parsing strategies has been used (Ivanov and Bobick 2000, Moore and Essa 2001, Johnston 2000). In these approaches the goal is to recognize structurally defined relationships of features where purely statistical approaches to recognition are less than ideal. These situations, can be characterized (following Ivanov and Bobick 2000) by one or more of the following conditions:

- *Insufficient data:* Complete data sets are not always available, but component examples are easily found.
- *Semantic ambiguity:* Semantically equivalent processes possess radically different statistical properties.
- *Temporal ambiguity:* Competing hypotheses can absorb different lengths of the input stream, raising the need for naturally supported temporal segmentation.
- *Known structure:* Structure of the process is difficult to learn but is explicit and a priori known.

When these conditions arise, it seems natural to divide the problem in two:

1. Recognition of low-level features (primitives) and
2. Recognition of structure.

The goal then becomes to combine statistical detection of primitives with a structural interpretation method that organizes the data.

In this work we adopt the method of (Ivanov and Bobick 2000) that combines mainly statistical techniques used to detect low-level features (primitives) of a more complex process structure. We combine results of the lower level feature recognizers into a consistent interpretation with the maximum likelihood using a Probabilistic Context-Free Grammar (PCFG) parser. The grammar provides a convenient means for encoding the external knowledge about the problem domain, expressing the expected structure of the high-level process.

3 Overall architecture

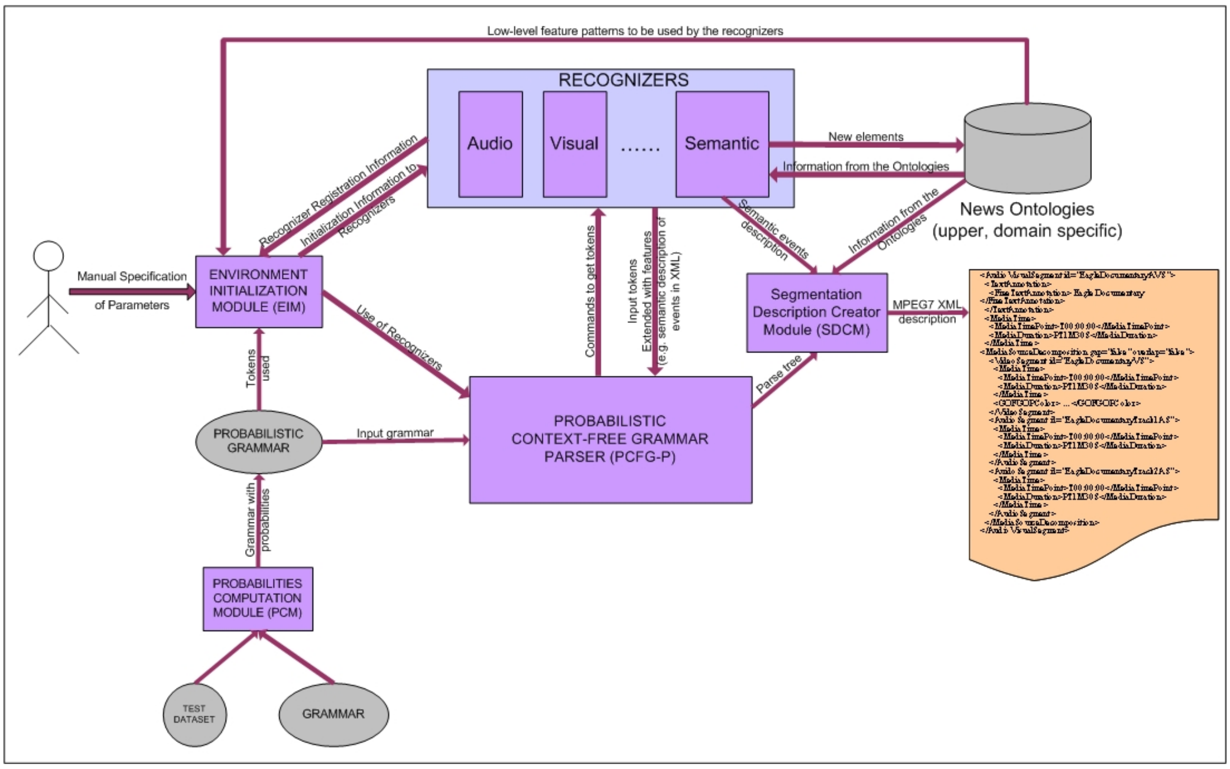
The architecture adopted is depicted in Figure 1. The central components of the architecture are the two main interoperating modules: the recognizers and the parser.

The recognizers analyze the telecast and each one identifies hypothesized instances of features in the audiovisual input. Such features could range from upper-level concepts like a specific person appearing in a shot (e.g., the anchor), the appearance of a certain series of frames (e.g., the introduction sequence, which many news broadcasts have in their beginning), to low-level features, e.g., the similarity of two frames.

The system includes three distinct recognizer modules. The audio recognizer, the visual recognizer and the semantic recognizer:

- The visual recognizer identifies visual features on the news stream, such as a face appearing at an expected position in the video or the presence of a familiar frame according to the expected structure of the broadcast.
- The audio recognizer, in turn, identifies audio features such as the presence of speech or music, e.g., a signature tune, and clustering of speakers.
- Finally, the semantic recognizer identifies the semantics involved in the telecast. This includes topic detection, high-level event detection, discourse cues as well as possible story segmentation points.

The PCFG (Probabilistic Context-Free Grammar) parser interoperates with the recognizers. The parser uses a probabilistic grammar in order to analyze the identifications provided by the recognizers. In essence, the recognizers provide the parser with actual lexical tokens such as a lexical analyzer would provide to a programming language parser. The grammar represents the possible structures the news telecast may have, so the parser can identify the exact structure of this telecast. The parser passes to the recognizer token probabilities (based on grammar rules probabilities and depending on the current status of the parsing procedure), which are used by the recognizer to aid in the recognition of lexical tokens, closing the feedback loop between the recognizer and the parser. When the parsing is complete and all the structural elements of the input have been analyzed, the Segmentation Description Creator Module uses that information to provide an MPEG7 compliant XML document with the description of identified news story segments with semantic topics and semantic events.


Low-level feature patterns to be used by the recognizers
RECOGNIZERS
Audio
Visual
……
Semantic
New elements
Information from the Ontologies
News Ontologies
(upper, domain specific)
Recognizer Registration Information
Initialization Information to Recognizers
Manual Specification of Parameters
ENVIRONMENT INITIALIZATION MODULE (EIM)
Use of Recognizers
Commands to get tokens
Input tokens Extended with features (e.g. semantic description of events in XML)
Semantic events description
Information from the Ontologies
Segmentation Description Creator Module (SDCM)
MPEG7 XML description
Parse tree
Tokens used
PROBABILISTIC GRAMMAR
Input grammar
PROBABILISTIC CONTEXT-FREE GRAMMAR PARSER (PCFG-P)
Grammar with probabilities
PROBABILITIES COMPUTATION MODULE (PCM)
TEST DATASET
GRAMMAR

```
<AudioVisualSegment id="EagleDocumentaryAVS">
 <TextAnnotation>
  <FreeTextAnnotation> Eagle Documentary
</FreeTextAnnotation>
 </TextAnnotation>
 <MediaTime>
  <MediaTimePoint>T00:00:00</MediaTimePoint>
  <MediaDuration>PT1M30S</MediaDuration>
 </MediaTime>
<MediaSourceDecomposition gap="false" overlap="false">
  <VideoSegment id="EagleDocumentaryVS">
   <MediaTime>
    <MediaTimePoint>T00:00:00</MediaTimePoint>
    <MediaDuration>PT1M30S</MediaDuration>
   </MediaTime>
   <GOFGOPColor> ... </GOFGOPColor>
  </VideoSegment>
  <AudioSegment id="EagleDocumentaryTrack1AS">
   <MediaTime>
    <MediaTimePoint>T00:00:00</MediaTimePoint>
    <MediaDuration>PT1M30S</MediaDuration>
   </MediaTime>
  </AudioSegment>
  <AudioSegment id="EagleDocumentaryTrack2AS">
   <MediaTime>
    <MediaTimePoint>T00:00:00</MediaTimePoint>
    <MediaDuration>PT1M30S</MediaDuration>
   </MediaTime>
  </AudioSegment>
 </MediaSourceDecomposition>
</AudioVisualSegment>
```

Figure 1: Overall architecture

The grammar for each broadcast station, even for different news programs of the same station, is distinct. This is because the grammar captures the directing elements of the broadcast, and no two different programs have exactly the same directional structure. Therefore, an initial context-free grammar (without probabilities) has to be produced manually for each program of interest, a task on which the system relies heavily, if processing is to be robust.

For the grammar to be probabilistic, it is necessary to complete a training process in order to assign probabilities to each production rule in the grammar. This training is carried out by the Probabilities Computation Module which uses a set of correctly labeled broadcast scenarios (in the form of a sequence of tokens). The examples can either be created manually, or through a semi-automatic annotation tool for ease of use.

The semantic recognizer has access to the Upper Ontologies as well as domain specific ontologies created for news. The concepts acquired from these ontologies will define those detectable semantics that can be identified in the telecast. It should be noted here the semantic recognizer relies on the news telecast's transcript to function. The current version of the audio recognizer does not provide the transcript. However, the test dataset used (TRECVID '03 collection) does provide such transcripts and are used by the current version of the semantic recognizer. In future versions the audio recognizer will provide to the semantic recognizer the necessary transcripts.

The Environment Initialization Module provides a basic system initialization, according to the "working environment", a concept that encapsulates specific parameters of the system functionality, according to the input that will be analyzed.

4 Main Components of the Architecture

The visual recognizer, the audio recognizer and the semantic recognizer modules generate input for the probabilistic parser in the form of visual, audible and semantic tokens that are contained in news video, making it possible to recognize the video’s inherent structure. The three recognizers are described in the following sections. This is followed by the descriptions of the modules related to the PCFG parsing and the generation of the final MPEG7 compliant segmentation and semantic description of the news videos. An additional application of the visual and audio recognizers has been done in TRECVID 2006, with focus on the detection of so-called high-level features instead of structural tokens (Bruckmann *et alii* 2006).

4.1 Visual Recognizer

The image feature classifier uses every 20th frame of an input video to do its calculations. Within these calculations filters are used to calculate a probability for every high level feature/ structural token. The following visual features are computed on every considered frame, followed by an SVM-based classification to get a visual token.

4.1.1 Color correlogram

This filter is an implementation of a standard color correlogram filter (Huang *et alii* 1997). A color correlogram (henceforth correlogram) expresses how the spatial correlation of pairs of colors changes with distance. We used the correlogram for the high-level features Charts, Desert, Explosion/Fire, and Maps.

4.1.2 Color histogram filter

The color histogram filter reduces the amount of colors by removing a parameterizable number of bits inside every RGB color channel. We use color histograms with sizes of 4 or 8 bins per dimension (R, G, and B), for a total of 64 or 512 bins, respectively. These histograms are used either with or without the ROI filter (see above).

4.1.3 Text detection filter

The text detection filter is based on an algorithm developed in a Diploma thesis at the University of Bremen (Wilkens 2003). It uses an edge detection filter tailored for overlaid text to find single characters and searches for regions where many characters appear on a line. It tries to find words and sentences by their typical spatial distribution and returns the positions of these inside an image. We adapted the algorithm's parameters to values we found to work good on an internal test set.

4.1.4 Edge direction histogram

This filter is an implementation of a standard edge direction histogram by H. Tamura *et alii* 1978. We use it to analyze an image for directed textures. Tamura's directionality criterion is characterized by a histogram based on eight orientations.

4.1.5 Image feature classification

We created two different sets for every feature/token to find, an internal training set and an internal test set. The training set was used for building classification models, while the internal test set was used to validate these models. We use a support vector machine (SVM) in the form of the SVM-light software (Joachims 1999) to train one model for each high-level feature/structural token, based on our image filter results as described in the previous section. As a kernel we chose the radial basis function (RBF) kernel. We set the relative significance of positive vs. negative examples to their relative frequency. In our validation process, we vary the variance parameter, and the trade-off between training error and margin. We validate a model created with given parameters on our internal test set, using the F-measure. We retain the model with the highest F-measure.

4.2 Audio Recognizer

The audio classifier searches the audio tracks of the input videos for a number of previously learned sounds. The classifier is built up of two stages. In the first stage we extract spectral features from the audio tracks. In the second step we use a supervised learning algorithm for training and prediction. Our approach is based on the algorithm proposed by Hoiem, Ke and Sukthankar 2005 but it differs in classifying. While Hoiem, Ke and Sukthankar 2005 suggest a decision tree classifier, we chose to use a support vector machine (SVM) in the classifier stage, in the form of the Libsvm library (Chang and Lin 2001). We will first describe the feature extraction step, followed by a description of our classification using a support vector machine.

4.2.1 Audio feature extraction

The first step of the sound feature extraction module is to create an abstract feature representation of the audio signal using an FFT on a temporally shifted window of the audio track. From the spectral representation, a set of 63 descriptive features is computed (Hoiem, Ke and Sukthankar 2005) to serve as input to the classifier. The size of the window is dependent of the type of sound that should be detected. The longer the sound, the bigger the window. We use windows ranging from 800 milliseconds to 1200 milliseconds. When applying the final classifiers to the test set, the window is shifted in steps of 100 milliseconds. For training, we manually cut a set of training sounds for each type of sound to be detected.

4.2.2 SVM classification

We chose to manually create a training set for the sound we wanted to detect. We cut a small number of short example sounds between 0.5 and 2.5 seconds, including all the disturbing sounds that might be in the background. The manual searching and cutting of sample sounds takes a long time, but is in the end the only way to ensure that the system learns the right type of sounds. It turned out that the selection type and also the number of training sounds has a great effect on the prediction quality of the SVM. During the testing of the system for various videos, the prediction sometimes returned very few or no results, even if the news contained plenty of the regarded sound events. The reason for this was probably that the analyzed sounds were too different from the training examples. However, finding good training examples that cover the whole variance of sounds is hard to manage, and it is very hard to cut the sounds from the test material. Our solution to this problem was to lower the threshold on the prediction values yielded by the SVM classifier, such that not only positive predictions are counted, but also negative results down to -0.4 or lower. That way, we reached a much bigger number of positives.

4.3 Semantic Recognizer

The semantic recognizer component is responsible for all semantic-related recognition and classification involved in the system. In a first simplified approach, the semantic recognizer is based on the format of the TRECVID'03 collection transcripts, and encloses mechanisms for a three-staged transcript and words in transcript processing.

At first stage the Semantic Recognizer briefly parses the transcript in order to extract and internally store the transcript words and other information. Except from the words spoken into the video, the transcript contains important information like the pauses that the speaker makes. Such information can be of significant importance when trying to define when a topic starts or ends. The pauses described in the transcript can even determine when the speaker emphasizes a word or phrase, giving valuable hints such as the significant keywords in a topic facilitating semantic event extraction.

The second stage of transcript processing concerns the tagging of the words previously found into the transcript. For this task a part-of-speech (POS)-tagger is involved (http://nlp.stanford.edu/software/tagger.shtml). The POS-tagger is responsible for the tagging of the words given to it in sentences. An important issue at this point was how to bind together words in sentences for which no information about what their relation is. A proper sentence-grouping is very important in order to achieve the most accurate tagging. For this reason a pre-tagging is made in order to locate subjects, verbs and objects. In addition, with the aid of the pauses described above, an estimation of sentence-grouping is given to the POS-tagger. This way, the POS-tagger is able to perform an appropriate word-tagging. After this the final parsing is made for the extraction of the words tagged as nouns in order to be used in the third stage.

The third and last stage of the processing of words in transcripts, targets on topic detection. The idea is to use lexical chains (Stokes, Carthy and Smeaton 2004) and lexical similarity upon the nouns into the transcript. In order to implement this it is necessary to define the window length for the lexical chaining and the measure of calculating the similarity of the words in each window. Test results until now give us 5 to 7 words as a satisfying window length spotting the expected limits of topics in the test data set. Similarity between the words in each window is currently calculated using WordNet (Miller *et alii* 1990).

Briefly, the whole procedure for topic detection starts with noun extraction from the video transcript. The similarity of the first nouns, whose number is defined by the window length, is calculated and stored. Then the window moves forward one noun and the similarity of the new window is calculated again and stored. The same procedure is followed until the window has passes upon all nouns; finally all the calculated similarities are compared. This comparison shows how the similarity between nouns goes up and down as the window passes from the first to the last noun in the transcript. Between the words that the similarity apparently decreases, the possibility of a topic change increases. This approach, combining the sentence-grouping described at stage two of parsing the transcript, and the pauses traced at stage one, provides a good estimation of where the topics start and end. This estimation is made available to the PCFG Parser.

In the next version of the semantic recognizer, actual ontologies will be used to facilitate better performance in the identification of semantic features as well as to perform semantic event extraction from news stories identified.

4.4 Probabilistic Parsing

As already stated, the temporal structure characteristic for a news broadcast format may be modeled as a probabilistic context free grammar (PCFG) with probabilities associated with production rules. This is a probabilistic extension of a Context-Free Grammar that is found to be very efficient in video recognition tasks (see for example Ivanov and Bobick 2000). The extension is implemented by adding a probability measure to every production rule:

$$a \rightarrow L \ [p]$$

This probability is a conditional probability of the production being chosen, given that non-terminal a is up for expansion (in generative terms). Saying that a probabilistic grammar is context-free essentially means that the rules are conditionally independent and, therefore, the probability of the grammar generating a particular complete derivation is simply the product of the probabilities of rules participating in the derivation.

In order to integrate story segmentation and classification information, the scenario PCFG is based on the news segmentation model that contains all the different parts of a news story and news broadcast such as: INTRO, START-OF-PREVIEW, PREVIEW, END-OF-PREVIEW, START-OF-STORY, PRESENTATION, REPORT, END-OF-STORY, COMMERCIAL, OUTRO etc. These structural components may be inferred by the PCFG parser from tokens coming from the recognizer modules using appropriate grammar rules.

4.4.1 Implementation of the Parsing Process

The implementation of the PCFG parsing process distinguishes between two necessary modules:

1. The PCFG parser component that receives a PCFG describing the structure of a specific type of a news telecast, and
2. An Environment Initialization module that provides initialization parameters to the system including the initialization of the recognizers with respect to the PCFG used.

4.4.1.1 The PCFG parser

The PCFG parser uses the Probabilistic Extension of the Earley Parsing Algorithm, based on a paper by Stolcke 1995. The key reasons for this decision are:

- The algorithm performs context-free parsing with probabilities computation.
- The Earley algorithm is one of the most efficient parsing algorithms, with complexity O(n), where n is the total number of the production rules in the grammar.
- The prediction step of the algorithm gives to the parser the opportunity to have an overall control of the recognizers.

Earley parser is a combined top/down bottom/up parsing algorithm that spans the parse tree on both directions. It has been proved that its performance is at worst O(n) (Earley 1968). To be more accurate, rather than using a span tree, the Earley parser uses the concept of a chart, where every edge represents a production rule.

The algorithm proceeds in three steps:

- *Prediction:* The prediction step is used to hypothesize the possible continuation of the input based on the current position in the parse. Prediction essentially expands one branch of the grammar down to the set of its leftmost leaf nodes to predict the next possible input terminals. At this step, any edge corresponding to predicted production rules is added to the chart as a pending (or active) edge.
- *Scanning:* Scanning step simply reads an input symbol and matches it against all pending states for the next iteration. When a pending state is matched, a terminal rule is added to the chart, and the pending state becomes inactive.
- *Completion:* The completion step updates pending derivations. When a production rule has finished, it converts the corresponding edge to a completed one.

When the parsing is finished, the algorithm walks through the chart and exports the parsing tree from the production rules that have been completed.

The probabilistic extension of the Earley parsing algorithm uses a PCFG, i.e. a context-free grammar with probabilities attached to each production rule. During parsing, the algorithm computes two kinds of probability metrics:

- A forward probability, which represents the probability of the whole parse tree.
- An inner probability, which represents the probability for every individual parse step.

The forward probability can be used to choose the most probable parse among many completed parses and the inner probability is used for clever pruning, where we can define probability thresholds and reduce the size of the parse tree accordingly.

The ambiguity of the input tokens due to the probabilistic nature of the recognizers affects computed probabilities at the scanning step of the algorithm. In this way, the values of inner and forward probabilities will weight competing derivations not only by the typicality of corresponding production rules, but also by the certainty about the input symbol at each step.

We have introduced some modifications to the parsing algorithm described above. One major change is the ability to add many candidate input tokens to the chart at every scanning step (note that the probabilistic Earley algorithm described above adds only one token symbol at every scanning step). This modification was necessary to address the need to use an arbitrary number of recognizers working in synch and each producing a series of tokens along with their likelihood. Some of these candidate symbols can be erroneous, but this does not represent a problem for our implementation, because candidate symbols not leading to further rule productions are discarded and do not affect the continuation of parsing.

Another important modification is the use of the prediction step of the algorithm so as to guide the recognizers. The prediction step of the algorithm gives the parser the opportunity to have an overall control of the recognizers, by giving them information of the predicted forthcoming tokens. In this way, we achieve overall system efficiency, by letting recognizers search for particular tokens in a specific part of the audio-visual input.

Another major aspect is the aspect of time and its importance in the overall parsing process. Parsing of input works time-incrementally. At every parsing cycle, parser tracks the duration of the input parsed. Parsing advances for the remaining of input and recognizers return tokens from a specific time point and for a specific duration. Time constraints also affect parsing at the completion step of the algorithm, where some active rules (edges) are discarded if their time duration exceeds current time limits of parsing. The concept of time helps us reduce error productions resulting from error recognitions from recognizers, providing a means for error correction. The time concept is encapsulated in the probabilistic grammar, where every rule includes a time-duration feature. Moreover the time information, as accumulated during parsing, provides the means to make the actual segmentation of the telecast identifying where each segments starts and ends.

For the implementation of the parser we have used Javachart parser (http://nlpfarm.sourceforge.net/javachart/) as our implementation framework. The Javachart parser implements a bottom-up chart parsing algorithm and uses features for every edge in the chart. In our implementation we altered the main parsing algorithm to function as a probabilistic Earley parser.

4.4.1.2 Environment Initialization

The Environment Initialization Module will provide system initialization, according to the "working environment". The working environment encapsulates specific parameters of the system functionality, according to the audio-visual input. For example, analyzing CNN's news telecast will require to use different parameters from analyzing BBC news (the appropriate images representing the ANCHORMAN_TOKEN is different for these two examples). The Environment Initialization Module will parameterize recognizers, may give appropriate information to the parser about which recognizers are available for every telecast analysis, or it can "map" tokens between the grammar and the recognizers by simply informing recognizers about the actual data that the tokens consist of (e.g. giving information to the video recognizer that the ANCHORMAN_TOKEN is the appearance of an instance of the image of John Foo). The module may also provide a user interface for manual initialization of working parameters.

For advanced functionality, the module must use already developed ontologies that describe specific low level features. (e.g., the sample images or voice patterns of the possible newscasters in a news telecast) in order to pass this information to the recognizers for appropriate feature extraction from the audiovisual signal.

4.4.2 Training the Probabilistic Grammar

The training of the grammar will be performed by parsing "perfect" tokenizations of a set of news telecasts. As in traditional feature grammar training methods, the number of times each production rule is expanded the higher is its final probability measure. The algorithm that will be employed is described in (Ivanov and Bobick 2000).

The "perfect" token representations of a telecast can either be created manually or with the assistance of a semi-automated annotation tool, where the user is called to correct the output of tokens.

4.5 Segmentation Description Creator Module

The Segmentation Description Creator Module (SDCM) will create the final output of the system, an MPEG7 compliant XML document with the segmentation and semantic information metadata. In order to produce this XML document, the most probable parse tree of the parsing phase will be used along with the information of the detected semantic events from the semantic recognizer. The News Ontologies will also be used in order to find additional information regarding the semantic content of the detected segments (i.e. news stories).

5 The parser in detail and an example of its operation and the communication with the recognizers

5.1 The parsing algorithm in detail

Before describing the overall operation of the algorithm, we will give the basic notation used later on. The input string is denoted as x. |x| is the length of x. Individual input symbols are identified by indices starting at 0: $x_0,\ldots,x_{|x|-1}$. The input alphabet is denoted as Σ. Substrings are identified by beginning and end positions $x_{i,\ldots j}$. The capital letters X, Y, Z … denote non-terminal symbols. Latin lowercase letters a, b, c … are used for terminal symbols. Strings of mixed non-terminals and terminals symbols are written using Greek letters λ, μ, ν … The empty string is denoted by ε.

An Earley parser builds left-most derivations of strings of tokens based on a given set of context-free grammar rules. It keeps track of all possible derivations that are consistent with the input string up to a certain point. As the input gets processed the set of possible derivations, each corresponding to a parse, can either expand as new choices are introduced, or shrink as a result of resolved ambiguities.

The parser keeps a set of states for each position in the input, describing all pending derivations. These states sets together form the Earley chart. A state is of the form:

$$i : {}_kX \rightarrow \lambda.\mu$$

where X is a non-terminal of the grammar, λ and μ are strings of non-terminals and/or terminals, and i and k are indices into the input string. States are derived from productions in the grammar. The above state is derived from a corresponding production

$$X \rightarrow \lambda\ \mu$$

with the following semantics:

- The current position in the input is i i.e. $x_0,\ldots,x_{i-1}$ have been processed so far. The states describing the parser state at position i are collectively called *state set i*. Note that there is one more state set than input symbols: set 0 describes the parser state before any input is processed, while set |x| contains the states after all input symbols have been processed.
- Non-terminal X was expanded starting at position k in the input, i.e., X generates some substring starting at position k.
- The expansion of X preceded using the production X → λμ, and has expanded the right-hand side (RHS) λμ up to the position indicated by the dot. The dot thus refers to the current position i.

A state with the dot to the right of the entire RHS is called a *complete state*, since it indicates that the left-hand side (LHS) non-terminal has been fully expanded.

The operation of the parser is defined in terms of the three operations (prediction, scanning and completion) that consult the current set of states and current input symbol, and add new states to the chart. The three types of transitions operate as follows:

- *Prediction:* For each state

  $$i : {}_kX \rightarrow \lambda.Y\mu$$

  where Y is a non-terminal anywhere in the RHS, and for all rules Y→v expanding Y, add states

  $$i : {}_iY \rightarrow .v$$

  A state produced by prediction is called a *predicted state*. Each prediction corresponds to a potential expansion of a non-terminal in a left-most derivation.
- *Scanning:* For each state

  $$i : {}_kX \rightarrow \lambda.\alpha\mu$$

  where α is a terminal that matches the current input x_i, add the state

  $$i + 1: {}_kX \rightarrow \lambda\alpha.\mu$$

  (move the dot over the current symbol). A state produced by scanning is called a *scanned state*. Scanning ensures that the terminals produced in a derivation match the input string.
- *Completion:* For each complete state

  $$i : {}_jY \rightarrow v.$$

- and each state in set j, j<=i, that has Y to the right of the dot,

  $$j: {}_kX \rightarrow \lambda.Y\mu$$

  add the state

  $$i : {}_kX \rightarrow \lambda Y.\mu$$

  (move the dot over the current non-terminal). A state produced by completion is called a *completed state*. Each completion corresponds to the end of a non-terminal expansion started by a matching prediction step.

To complete the description we need only specify the initial and final states. The parser starts out with:

$$0:\ 0 \rightarrow .S$$

where S is the start symbol of the grammar. After processing the last symbol, the parser verifies that:

$$l : 0 \rightarrow S.$$

has been produced (among possible others), where l is the length of the input x. When the parsing is finished, the algorithm walks through the chart and exports the parsing tree from the production rules that have been completed (complete states).

The probabilistic Earley parsing algorithm operates on PCFGs. As mentioned before the probabilistic extension of the Earley algorithm computes inner and forward probabilities for every edge in the chart. Forward probability represents the likelihood of the parsed input from position 0 and up to current position whereas inner probability represents the likelihood of the expansion of the input from a given non-terminal – in other words from a position k – and up to current position. During a run of the parser both probabilities are attached to each state and updated incrementally as new states are created through every one of the three types of transitions.

The forward and inner probabilities of states are denoted in brackets after each state, e.g.,

$$i : {}_kX \rightarrow \lambda.Y\mu \ \ [\alpha,\gamma]$$

is shorthand for $\alpha = \alpha_i\,({}_kX\rightarrow \lambda.Y\mu)$, $\quad \gamma = \gamma_i\,({}_kX\rightarrow \lambda.Y\mu)$

For every step (prediction, scanning, completion), the algorithm performs additional accumulations (Ivanov and Bobick 2000, Stolcke 1995):

- *Prediction* (probabilistic)

$$i : {}_kX\rightarrow \lambda.Z\mu \ \ [\alpha,\gamma] \ \Rightarrow \ i :{}_iY\rightarrow .v \ \ [\, \alpha' , \gamma' \,]$$

for all productions $Y \rightarrow v$. The new probabilities can be computed as

$$\alpha' \ {+}{=} \ \alpha\, R(Z \overset{*}{\Rightarrow}_L Y)\, P(Y\rightarrow v)$$
$$\gamma' = P(Y\rightarrow v)$$

The $R(Z \overset{*}{\Rightarrow}_L Y)$ factor is an updated forward probability that accounts for the sum of all paths probabilities linking Z to Y.

- *Scanning* (probabilistic)

$$i : {}_kX\rightarrow \lambda.\alpha\mu \ \ [\alpha,\gamma]$$
$$\Rightarrow \ i + 1: {}_kX\rightarrow \lambda\alpha.\mu\ , \ \ [\alpha',\gamma']$$

for every α, $P(\alpha)>0$

New values of α' and γ' are:

$$\alpha' = \alpha\ (\ i : {}_kX \rightarrow \lambda.\alpha\mu\ )\ P(\alpha)$$
$$\gamma' = \gamma\ (\ i : {}_kX\rightarrow \lambda.\alpha\mu\ )\ P(\alpha)$$

The ambiguity of the input tokens due to the probabilistic nature of the recognizers affects computed probabilities at the scanning step of the algorithm. In this way, the values of inner and forward probabilities will weight competing derivations not only by the typicality of corresponding production rules, but also by the certainty about the input symbol at each step

- *Completion* (probabilistic)

$$\begin{array}{l} i : {}_jY\rightarrow v. \ \ [\alpha'',\gamma''] \\ \qquad\qquad\qquad\qquad \Rightarrow \ \ i : {}_kX\rightarrow \lambda Z.\mu \ \ [\alpha',\gamma'] \\ j : {}_kX\rightarrow \lambda.Z\mu \ \ [\alpha,\gamma] \end{array}$$

for all Y, Z such that $R(Z \overset{*}{\Rightarrow} Y)$ is nonzero and $Y\rightarrow v$ is not a unit production, then

$$\alpha' \ {+}{=} \ \alpha\, \gamma''\, R(Z \overset{*}{\Rightarrow} Y)$$

$$\gamma' \ {+}{=} \ \gamma\, \gamma''\, R(Z \overset{*}{\Rightarrow} Y)$$

where $R(Z \overset{*}{\Rightarrow} Y)$ is a probabilities relation used to collapse all unit completions in a single step.

5.3 Example for the operation of the parser

Consider a simple news telecast that starts with a news story followed by (a) a weather forecast and a financial news story or (b) a commercial and a second news story. A news story consists of a presentation (the anchor presents the story) and a report (the main part of the news story). A financial story contains an introduction and the story (consisting again of a presentation and a report just as an ordinary news story. A presentation is found when the video recognizer returns a token signifying the appearance of the anchor's face. A report is recognized when the audio recognizer finds speech in the audio signal or the video recognizer finds some text caption or the semantic recognizer finds a topic change. A commercial is found when some black frames appear in the telecast. The weather forecast segment is characterized by a map with temperatures (found by the video recognizer). The introduction to a financial story is spotted by the dollar sign frames or by the appearance of music in the audio signal. This news telecast structure can be captured by the following grammar (non-terminal symbols start with an upper case letter, terminals start with lower case letters):

S → Story SecondPart	[1.0]
Story → Presentation Report	[1.0]
SecondPart → Commercial Story	[0.4]
SecondPart → Weather FinancialStory	[0.6]
FinancialStory → FinancialIntro Story	[1.0]
Presentation → v_anchor_face	[1.0]
Report → a_speech	[0.2]
Report → v_text	[0.3]
Report → s_topic_change	[0.5]
Commercial → v_black_frames	[1.0]
Weather → v_temperature_map	[1.0]
FinancialIntro → v_dollar_sign	[0.7]
FinancialIntro → a_music	[0.3]

Note that in the grammar we have included special production rules (called *token production rules*) to capture how the different parts of news telecast are produced from specific tokens of the audio, video and semantic recognizers. Tokens coming from the audio recognizer start with a_ while tokens from the video recognizer start with v_ and tokens from the semantic recognizer start with s_. The token production rules of the grammar essentially model how a presentation, a report, a commercial, a weather forecast and an introduction to a financial story are recognized.

Using the above grammar (we omit probabilities' computation for simplifying the example) the parser start with the following state set:

initial state:	0: $_0$ → .S
predicted states:	0: $_0$S → .Story SecondPart
	0: $_0$Story → .Presentation Report
	0: $_0$Presentation → .v_anchor face

At this point the parser predicts (see predicted token productions above) that the next token should be v_anchor face and calls the video recognizer. Let us assume that the video recognizer finds this specific token and returns it to the parser. The parser enters state set 1 with the following states:

scanned states:	1: $_0$Presentation → v_anchor_face.
completed states:	1: $_0$Story → Presentation.Report
predicted states:	1: $_1$Report → .a_speech
	1: $_1$Report → .v_text
	1: $_1$Report → .s_topic_change

Now the parser predicts (see predicted token productions above) that the next token could be a_speech from the audio recognizer signifying that this part of the broadcast contains speech in the audio signal or v_text from the video recognizer signifying that there is some text caption in the frames of the video or s_topic_change from the semantic recognizer signifying a topic change. The parser calls the three recognizers passing to each one the expected token to be seen next. Let us assume that the video recognizer does not find any text caption nor the semantic recognizer finds any topic change because this is the first story in the broadcast. The audio recognizer succeeds in finding speech in the audio signal and the a_speech token is returned and processed next. This results in the following state set 2:

scanned states:	2: $_1$Report → a_speech.
completed states:	2: $_0$Story → Presentation Report.
	2: $_0$S → Story.SecondPart
predicted states:	2: $_2$SecondPart → .Commercial Story
	2: $_2$SecondPart → .Weather FinancialStory
	2: $_2$Commercial → .v_black_frames
	2: $_2$Weather → .v_temperature_map

The predictions made now refer to the video recognizer: Tokens v_black_frames, signifying a commercial and v_temperature_map, signifying a weather report segment are expected. These two predicted tokens are passed to the video recognizer. Let us assume that the video recognizer succeeds in finding a series of black frames so it returns to the parser the identified v_black_frames token. The state set 3 becomes:

scanned states:	3: $_2$Commercial → v_black_frames.
completed states:	3: $_2$SecondPart → Commercial.Story
predicted states:	3: $_3$Story → .Presentation Report
	3: $_3$Presentation → .v_anchor face

The parser predicts that the next token should be v_anchor face and calls the video recognizer. Let us assume that the video recognizer finds this specific token and returns it to the parser. The parser enters state set 4 with the following states:

scanned states:	4: $_3$Presentation → v_anchor_face.
completed states:	4: $_3$Story → Presentation.Report
predicted states:	4: $_4$Report → .a_speech
	4: $_4$Report → .v_text
	4: $_4$Report → .s_topic_change

Now the parser predicts that the next token could be a_speech from the audio recognizer or v_text from the video recognizer or s_topic_change from the semantic recognizer. The parser calls the three recognizers passing to each one the expected token to be seen next. Let us assume that the video recognizer does not find any text caption nor the audio recognizer finds any speech. However, the semantic recognizer succeeds in finding a topic change and the s_topic_change token is returned and processed next. Let us further assume that no additional processing of the broadcast is necessary and that this is the final token processed. This results in the following state set 5:

scanned states:	5: $_4$Report → s_topic_change.
completed states:	5: $_4$Story → Presentation Report.
	5: $_3$SecondPart → Commercial Story.
	5: $_0$S → Story SecondPart.
	5: $_0$ → S. (final state)

Having reached state 5: $_0$ → S. and taking into account that the length of the input processed is 5, we have reached the desired final state and the input is accepted as valid. Tracing back the appropriate complete states it is possible to reconstruct the parse tree pertaining to the specific input processed. From the grammar symbols in this parser tree we can logically segment the broadcast. This parse tree is shown in the following figure:

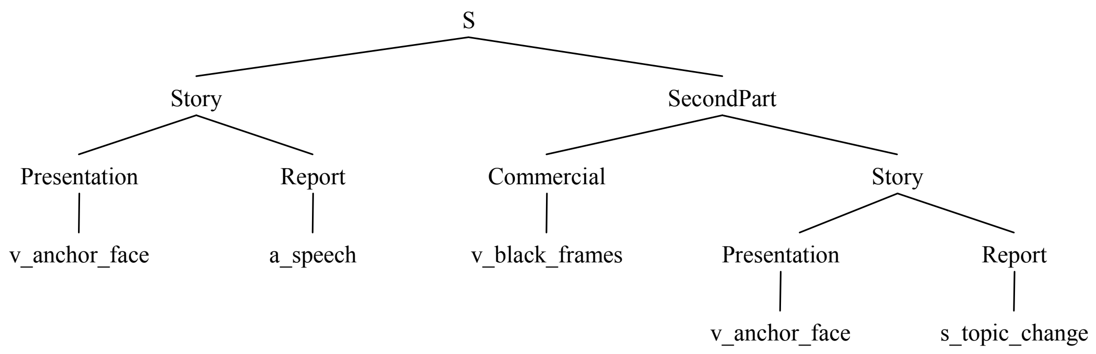


Figure 2 – An example parse tree resulting from the operation of the parser.

It is apparent from the above parse tree that the telecast processed contains two stories separated by a commercial. Note here that in order to achieve a realistic segmentation it necessary that the parser receives and processes time information from the recognizers representing the starting time and the finishing time for every token returned. It is also necessary that the production rules handle this time information appropriately in order to accumulate it as the parse tree is produced. For simplicity we have not included these computations in the example presented here.

5. Conclusions and Future Work

The implementation of the system described in the previous sections has not been finalized yet. The current implementation consists of separate recognizer modules along with the basic modules for parsing. In order to demonstrate the parsing process the parser operates on a simulated input (tokens) from the recognizers as shown in the following figure.

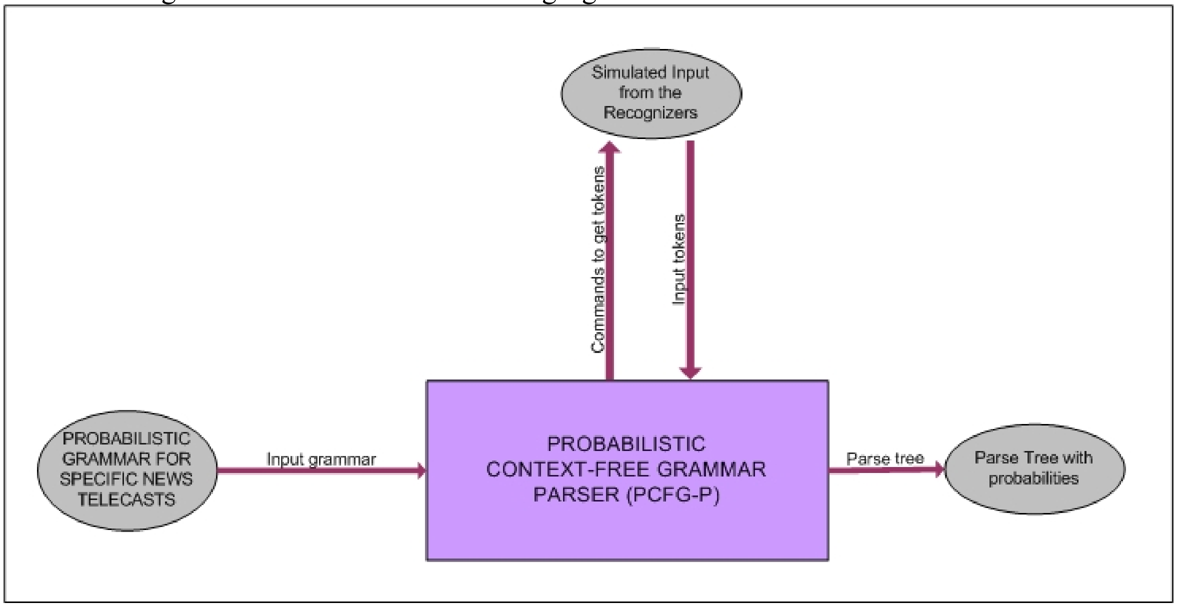

Simulated Input from the Recognizers
Commands to get tokens
Input tokens
PROBABILISTIC GRAMMAR FOR SPECIFIC NEWS TELECASTS
Input grammar
PROBABILISTIC CONTEXT-FREE GRAMMAR PARSER (PCFG-P)
Parse tree
Parse Tree with probabilities

Figure 3 – The demonstrator of the parser operating on simulated input.

In the final demonstrator, a full version of the Semantic Recognizer along with the fully developed News Ontologies and the Semantic Description Creator Module will be used to provide the final MPEG7 compliant XML description of the segmented and semantically analyzed news telecasts.

The current parser demonstrator provides a fully operational PCFG parser that uses available probabilistic grammars describing specific classes of news telecasts. These probabilistic grammars have been manually specified based on observations for the structure of classes of news telecasts available in the TRECVID'03 collection. Simulated streams of tokens, as they will be provided by the system's recognizers, have also been specified taking into account the capabilities of the recognizers and the grammars used. This simulated input is given to the PCFG parser for parsing. After parsing this simulated input, the parser returns with a set of parse trees along with their probabilities. The parse trees contain all the required information (in their inner nodes representing the non-terminals of each probabilistic grammar) for the segmentation of the news telecasts that have been used to produce each simulated stream of tokens.

The integration of the actual recognizers in this demonstrator requires the transformation of their current output format into token-like form as well as the provision of their functionality using a well defined web-services interface. This is currently under implementation.

References

Biatov, K. and Larson, M. 2005. Speaker clustering via Bayesian Information Criterion using a Global Similarity Constraint. In *Proceedings of Specom2005, 10th International Conference on Speech and Computers. 2005.*

Biatov, K., Larson, M. and Eickeler, S. 2002. Zero-crossing based temporal segmentation and classification of audio signals. In *Proceedings of the Sixth All-Ukrainian International Conference on Signal/Image Processing and Pattern Recognition. 2002.*

Boreczky J. S. and Wilcox L. D. 2000. A Hidden Markov Model Framework for Video Segmentation Using Audio and Image Features. In *proc. IEEE ICASSP. Seattle (USA), M ay 1998.*

Brand M., Kettnaker V. 2000. Discovery and Segmentation of Activities in Video, In *IEEE Trans. Pattern Anal. Mach. Intel,* 22(8):844-851.

Bruckmann A., Lerbs B., Gao D., Eidtmann J., Mozogovenko L., Buczilowski M., Jughardt T., Xu Y., Jacobs A. and Lüdtke A. 2006. Trecvid 2006 high level feature extraction. In *TRECVID 2006 Workshop Notebook Papers. 2006*.

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

Dimitrova N., Agnihotri L. and Wei G. 2000. Video classification based on HMM using text and faces. In *European Signal Processing Conference. Tampere (Finland), 2000.*

Earley J.C. 1968. *An Efficient Context-Free Parsing Algorithm.* PhD thesis. Carnegie-Mellon Univ.

Eickeler S. and Muller S. 1999. Content-based video indexing of TV broadcast news using hidden markov models. In *IEEE International Conference on Acoustics, Speech, and Signal Processing. Phoenix (USA), 1999.* 2997–3000.

Greiff W., Morgan A., Fish R., Richards M. and Kundu A. 2001. Fine-Grained Hidden Markov Modeling for Broadcast-News Story Segmentation. In *proceedings of the first international conference on Human language technology research. San Diego (USA), 2001.* 1-5.

Hoiem D., Ke Y. and Sukthankar R. 2005. Solar: Sound object localization and retrieval in complex audio environments. In *proc. of the IEEE International Conference on Acoustics, Speech and Signal. 2005.*

Huang J., Kumar R., Mitra M., Zhu W. J. and Zabih R. 1997. Image indexing using color correlograms. In *proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97).* 762.

Huang J., Liu Z., Wang Y., Chen Y. and Wong E. K. 1999. Integration of multimodal features for video scene classification based on HMM. In *IEEEWorkshop on Multimedia Signal Processing, Copenhagen (Denmark), 1999.*

Ivanov, Y. and Bobick, A. F. 2000. Recognition of Visual Activities and Interactions by Stochastic Parsing. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8).
Joachims T. 1999. *Making large-Scale SVM Learning Practical.* MIT-Press.

Johnston M. 2000. Deixis and Conjunction in Multimodal Systems. In *proceedings of the 18th conference on Computational linguistics.- Volume 1, 2000.* 362-368.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D. and Miller, K.J. 1990. Introduction to WordNet: an on-line lexical database. In *International Journal of Lexicography* 3(4): 235-244.

Moore D. and Essa I. 2001. Recognizing multitasked activities using stochastic context-free grammar. In *proceedings of Workshop on Models vs Exemplars in Computer Vision. 2001.*

Stokes N., Carthy J. and Smeaton A. F. 2004. SeLeCT: A lexical Cohesion Based News Story Segmentation System. In *Journal of AI Communications* 17(1): 3-12.

Stolcke A. 1995. An efficient Probabilistic Context-Free Parsing Algorithm That Computes Prefix Probabilities. In *Computational Linguistics* .21(2): 165-201.

Tamura H., Mori S., and Yamawaki T. 1978. Textural feratures corresponding to visual perception. In *IEEE Trans. Syst., Man, Cyb.* 8(6):460-473

Tsinaraki C., Polydoros P., Christodoulakis S. 2004. Integration of OWL ontologies in MPEG-7 and TVAnytime compliant Semantic Indexing. In *proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE). Riga (Latvia), June 2004*

Wilkens N. 2003. *Detektion von Videoframes mit Texteinblendungen in Echtzeit.* Diploma thesis. Universität Bremen.