Chapter 8

UNDERSTANDING THE SEMANTICS OF MEDIA

Malcolm Slaney, Dulce Ponceleon and James Kaufman
IBM Almaden Research Center
San Jose, California
malcolm@ieee.org

Abstract It is difficult to understand a multimedia signal without being able to say something about its semantic content or its meaning. This chapter describes two algorithms that help bridge the semantic understanding gap that we have with multimedia. In both cases we represent the semantic content of a multimedia signal as a point in a high-dimensional space. In the first case, we represent the sentences of a video as a time-varying semantic signal. We look for discontinuities in this signal, of different sizes in a one-dimensional scale space, as an indication of a topic change. By sorting these changes, we can create a hierarchical segmentation of the video based on its semantic content. The same formalism can be used to think about color information and we consider the different media's temporal correlation properties. In the second half of this chapter we describe an approach that connects sounds to semantics. We call this semantic-audio retrieval; the goal is to find a (non-speech) audio signal that fits a query, or to describe a (non-speech) audio signal using the appropriate words. We make this connection by building and clustering high-dimensional vector descriptions of the audio signal and its corresponding semantic description. We then build models that link the two spaces, so that a query in one space can be mapped into a model that describes the probability of correspondence for points in the opposing space.

Keywords: Segmentation, semantics, multimedia signal, high-dimensional space, video retrieval, audio analysis, latent semantic indexing (LSI), semantic content, clustering, non-speech audio signal, hierarchical segmentation, color information, topic change, sorting, temporal correlation, mixture of probability experts, MPESAR, SVD, scale space, acoustic space.

8.1 Semantic Understanding Problem

Due to the proliferation of personal cameras and inexpensive hard disk drives, we are drowning in media. Unfortunately, the tools we have to understand this media are very limited. In this chapter we describe tools that help us understand the meaning of our media. We will demonstrate that tools that analyze the semantic content are possible and this represents a high-level understanding of the media.

There are many systems which find camera shot boundaries—low-level events in the video where the camera changes to a new view of the scene [Srinivasan et al., 1999]. There are some tools, described below, which attempt to segment video at a higher level. But this level of analysis does not tell us much about the meaning represented in the media.

Only recently have researchers constructed higher-level understanding from multimedia signals. Aner [Aner and Kender, 2002] suggest an approach that finds the background in a video shot, and then clusters shots into physical scenes by noting shots with common backgrounds. This is one way to build up a higher-level representation of the video, but we argue that the most important information is in the words.

Retrieving media is a similarly hard problem. Systems such as IBM's QBIC system [Flickner et al., 1993] allow users to search for images based on the colors and images in an image. This is known as query-by-example, but most people don't think about their image requests in terms of colors or shapes. A better tool uses semantic information to retrieve objects based on the meaning in the media.

In the remainder of this section we will talk about specific approaches for segmentation and retrieval, and describe how our approaches differ. Section 8.1.3 describes the rest of this chapter.

8.1.1 Segmentation Literature

Our work extends previous work on text and video analysis and segmentation in several different ways.

Latent semantic indexing (LSI) has a long history, starting with Deerwester's paper [Deerwester et al., 1990], as a powerful means to summarize the semantic content of a document and measure the similarity of two documents. We use LSI because it allows us to quantify the position of a portion of the document in a multi-dimensional semantic space.

Hearst [Hearst, 1994] proposes to use the dips in a similarity measure of adjacent sentences in a document to identify topic changes. Her method is powerful because the size of the dip is a good indication of the relative amount of change in the document. We extend this idea using scale-space techniques to allow us to talk about similarity or dissimilarity over larger portions of the document.

Miller and her colleagues proposed Topic Islands [Miller et al., 1998], a visualization and segmentation algorithm based on a wavelet analysis of text documents. Their wavelets are localized in both time (document position) and frequency (spectral content) and allow them to find and visualize topic changes at many different scales. The localized nature of their wavelets makes it difficult to isolate and track segmentation boundaries through all scales. We propose to summarize the text with LSI and analyze the signal with smooth Gaussians, which are localized in time but preserve the long-term correlations of the semantic path.

Segmentation is a popular topic in the signal and image processing worlds. Witkin [Witkin, 1984] introduced scale-space ideas to the segmentation problem and Lyon [Lyon, 1984] extended Witkin's approach to multi-dimensional signals. A more theoretical discussion of the scale-space segmentation ideas was published by Leung [Leung et al., 2000]. The work described here extends the scale-space approach by using LSI as a basic feature and changing the distance metric to fit semantic data.

The key concept in our segmentation work is to think about a video signal's path through space, and detect jumps at multiple scales. The signal processing analysis proposed in this chapter is just one part of a complete system. We use a singular-value decomposition (SVD) to do the basic analysis, but more sophisticated techniques are also applicable. Any method which allows us to summarize the image and semantic content of the document can also be used in conjunction with the techniques described here.

8.1.2 Semantic Retrieval Literature

There are many multimedia retrieval systems that use a combination of words or examples to retrieve audio (and video) for users. Our algorithm, mixtures of probability experts for semantic-audio retrieval (MPESAR), is a more sophisticated model connecting words and media.

An effective way to find an image of the space shuttle is to enter the words "space shuttle jpg" into a text-based web search engine. The original Google system did not know about images, but, fortunately, many people created web pages with the phrase "space shuttle" and a JPEG image of the shuttle. The MPESAR work expands those search techniques by considering the acoustic and semantic similarity of sounds to allow users to retrieve sounds without running searches on the exact words used on the web page.

Barnard [Barnard and Forsyth, 2001] used a hierarchical clustering algorithm to build a model that combined words and image features to create a single hierarchical model that spanned both semantic and image features. He demonstrated the effectiveness of coupled clustering for an information-retrieval task and argued that the words written by a human annotator describing an image (e.g., "a rose") often provide information that complements the obvious information in the image (it is red).

MPESAR improves on three aspects of Barnard's approaches. First, the semantic and image features do not have the same probability distributions. Barnard's algorithm assumes that image features can be described by a multinomial distribution, while a Gaussian is probably more appropriate. Second, and perhaps most important, there is nothing in Barnard's algorithm that guarantees that the features used to build each stage of the model include both semantic and image features. Thus, the algorithm is free to build a model that completely ignores the image features and clusters the 'documents' based on only semantic features. Third, MPESAR interpolates between models. Previous work assigned each document to a single cluster and used a single model (winner-take-all) to map to the opposite domain. On the other hand, MPESAR calculates the probability that each cluster generates the query and then calculates a weighted average of models based on the cluster probabilities.

The MPESAR algorithm is appropriate for mapping one type of media to another. We illustrate the idea here using audio and semantic documents because audio retrieval is a simpler problem.

8.1.3 Overview

In this chapter, we describe semantic tools for understanding media[1]. The key to these tools is a representation of the media's content based on the words contained in the media, or words describing the media. In this work we use mathematical tools to represent a set of words as a point in a vector space. We will then use this vector representation of the semantic content to allow us to create a hierarchical table of contents for a multimedia signals, or to build a query-by-semantics system.

Our description of the semantic tools is structured as follows. In Section 8.2 of this chapter, we will describe some common tools and mathematics we use to analyze multimedia signals. Section 8.3 describes an algorithm for hierarchical segmentation that uses the color, acous-

[1]This chapter combines material first published elsewhere [Slaney et al., 2001; Slaney, 2002]

tic, and semantic information in the signal. Section 8.4 describes our semantic-retrieval algorithm, which is applied to audio retrieval.

8.2 Analysis Tools

We use two types of transformations to reduce raw text and video signals into meaningful spaces where we can find edges or events.

The SVD provides a principled way to reduce the dimensionality of a signal in a manner which is optimum, in a least-squared sense. In the next sub-sections, we describe how we apply the SVD to color and semantic information. The SVD transformation allows us to summarize different kinds of video data and combine the results into a common representation (Section 8.3.5).

8.2.1 SVD Principles

We express both semantic and video data as vector-valued functions of time, $\vec{x}(t)$. We collect data from an entire video and put the data into a matrix, $\mathbf{X}$, where the columns of $\mathbf{X}$ represent the signal at different times. Using an SVD, we rewrite the matrix $\mathbf{X}$ in terms of three matrices, $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{V}$, such that

$$\mathbf{X} = \mathbf{USV}^T. \tag{8.1}$$

The columns of the $\mathbf{U}$ and $\mathbf{V}$ matrices are orthonormal; $\mathbf{S}$ is a diagonal matrix. The values of $\mathbf{S}$ along the diagonal are ordered such that

$$S_{11} >= S_{22} >= S_{33} >= ... >= S_{nn} \tag{8.2}$$

where n is the minimum of the number of rows or columns of $\mathbf{X}$.

The SVD allows us to generate approximations of the original data. If the first k diagonal terms of $\mathbf{S}$ are retained, and the rest are set to zero, then the rank k approximation to $\mathbf{X}$, or $\mathbf{X}^k$, is the best possible approximation to X (in the least squares sense):

$$|\mathbf{X} - \mathbf{X}^k| = \min_{rank(\mathbf{Y}) \leq k} |\mathbf{X} - \mathbf{Y}| \geq |\mathbf{X} - \mathbf{X}_{k+1}|. \tag{8.3}$$

The first equality in equation 8.3 says that $\mathbf{X}^k$ is the best approximation in all k-dimensional subspaces. The second inequality states that, as we add more terms, and thus increase the size of the subspace, the approximation will not deteriorate (it typically improves).

Typically the first singular values are large; they then decay until a noise floor is reached. We want to keep the dimensions that are highly significant, while setting the dimensions that are dominated by noise to zero.

The columns of the **U** matrix are an ordered set of vectors that approximate the column space of the original data. In our case, each column of the **X** matrix is the value of our function at a different point in time. As we use more terms of **S**, the columns of **U** provide a better and better approximation to the cloud of data that forms from $\vec{x}(t)$.

Given the left-singular vectors **U** and our original data **X**, we project our data into the optimal k-dimensional subspace by multiplying

$$\mathbf{Y}^k = (\mathbf{U}^k)^T\mathbf{X} \tag{8.4}$$

where $\mathbf{U}^k$ contains only the first k columns of **U**, and $\mathbf{Y}^k = \vec{x^k}(t)$ is a k-dimensional function of time. We compute a new SVD and a new **U** matrix for each video, essentially creating movie-dependent subspaces with all the same advantages of speaker-dependent speech recognition.

We use the SVD to reduce the dimensionality of both our audio and image video data. The reduced representation is nearly as accurate as the original data, but is more meaningful (the noise dimensions have been dropped) and is easier to work with (the dimensionality is significantly lower).

8.2.2 Color Space

Color changes provide a useful metric for finding the boundary between shots in a video [Srinivasan et al., 1999]. We can represent the color information by collecting a histogram of the colors within each frame and noting the temporal positions in the video where the histogram indicates large frame-to-frame differences.

We collected color information by using 512 histogram bins. We converted the three red, green, and blue intensities— each of which range in value from 0 to 255— to a single histogram bin by finding the log, in base 2, of the intensity value, and then packing the three colors into a 9-bit number using floor() to convert to an integer:

$$Bin = 64\,\mathrm{floor}(\log_2(R)) + 8\,\mathrm{floor}(\log_2(G)) + \mathrm{floor}(\log_2(B)) \tag{8.5}$$

We chose this logarithmic scaling because it equalizes the counts in the different bins for our test videos.

The color histogram of the video frames converts the original video images into a 512-dimensional signal that is sampled at 29.97 Hz. The order of the dimensions is arbitrary and meaningless; the SVD will produce the same subspace regardless of how the rows or columns of the **X** matrix are arranged.

8.2.3 Word Space

Latent semantic indexing (LSI), a popular technique for information retrieval [Dumais, 1991], uses an SVD in direct analogy to the color analysis described above. As we did with the color data, we start analyzing the audio data by collecting a histogram of the words in a transcript of the video.

Normally, in information retrieval, each document is one of a large collection of electronically-formatted documents from which we want to retrieve the best match. In our case we want to study only a single document, so we consider portions of that document—sentences. The sentences of a document define a semantic space; each sentence, in general, represents a specific point in the semantic space.

Two difficult problems associated with semantic information retrieval are posed by synonyms and polysemy. Often, two or more words have the same meaning—synonyms. For information retrieval, we want to be able to use any synonym to retrieve the same information. Conversely, many words have multiple meanings—polysemy. For example, apple in a story about a grocery store is likely to have a different meaning from Apple in a story about a computer store.

The SVD allows us to capture both relationships. Words that are frequently used in the same section of text are given similar counts in the histogram. The SVD is sensitive to this correlation, in that one of the singular vectors points in the combined direction. Furthermore, words such as apple show up in two different types of documents, representing the two types of stories and will thus contribute to two different directions in the semantic space.

Changes in semantic space are based on angles, rather than on distance. A simple "sentence" such as "Yes!" has the same semantic content as "Yes, yes!" Yet the second sentence contains twice as many words, and, in semantic space, it will have a vector magnitude that is twice as large. Instead of using a Euclidean metric, we describe the similarity of two points in semantic space by the angle between the two vectors. We usually compute this value by finding the cosine of the angle between the two vectors,

$$cos(\phi) = (\nu_1 \cdot \nu_2)/(|\nu_1||\nu_2|). \tag{8.6}$$

8.3 Segmenting Video

Browsing videotapes of image and sound (hereafter referred to as "videos") is difficult. Often, there is an hour or more of material, and there is no roadmap to help viewers find their way through the medium.

It would be tremendously helpful to have an automated way to create a hierarchical table of contents that listed major topic changes at the highest level, with subsegments down to individual shots. DVDs provide the chapter indices; we would like to find the position of the sub-chapter boundaries. Realization of such an automated analysis requires the development of algorithms which can detect changes in the video or semantic content of a video as a function of time. We propose a technology that performs this indexing task by combining the two major sources of data—images and words—from the video into one unified representation.

With regard to the words in the sound track of a video, the information-retrieval world has used, with great success, statistical techniques to model the meaning, or semantic content, of a document. These techniques, such as LSI, allow us to cluster related documents, or to pose a question and find the document that most closely resembles the query. We can apply the same techniques within a document or, in the present case, the transcript of a video. These techniques allow us to describe the semantic path of a video's transcript as a signal, from the initial sentence to the conclusions. Thinking about this signal in a scale space allows us to find the semantic discontinuities in the audio signal and to create a semantic table of contents for a video.

Our technique is analogous to one that detects edges in an image. Instead of trying to find similar regions of the video, called segments, we think of the audio–visual content as a signal and look for "large" changes in this signal or peaks in its derivative. The location of these changes are edges; they represent the entries in a table of contents.

8.3.1 Temporal Properties of Video

The techniques we describe in this chapter allow us to characterize the temporal properties of both the audio and image data in the video. The color information in the image signal and the semantic information in the audio signal provide different information about the content.

Color provides robust evidence for a shot change in a video signal. An easy way to convert the color data into a signal that indicates scene changes is to compute each frame's color histogram and to note the frame-by-frame differences [Srinivasan et al., 1999]. In general, however, we do not expect the colors of the images to tell us anything about the global structure of the video. The color balance in a video does not typically change systematically over the length of the film. Thus, over the long term, the video's overall color often does not tell us much about the overall structure of the video.

Random words from a transcript, on the other hand, do not reveal much about the low-level features of the video. Given just a few words from the audio signal, it is difficult to define the current topic. But the words indicate a lot about the overall structure of the story. A documentary script may, for instance, progress through topic 1, then topic 2, and finally topic 3.

We describe any time point in the video by its position in an color–semantic vector space. We represent the color and the semantic information in the video as two separate vectors as a function of time. We concatenate these two vectors to create a single vector that encodes the color and the semantic data. Using scale-space techniques we can then talk about the changes that the color–semantic vector undergoes as the video unwinds over time. We label as segment boundaries large jumps in the combined color–semantic vector. "Large jumps" are defined by a scale-space algorithm that we describe in Section 8.3.4.

8.3.2 Segmentation Overview

This chapter proposes a unified representation for the audio–visual information in a video. We use this representation to compare and contrast the temporal properties of the audio and images in a video. We form a hierarchical segmentation with this representation and compare the hierarchical segmentation to other forms of segmentation. By unifying the representations we have a simpler description of the video's content and can more easily compare the temporal information content in the different signals.

As we have explained, we combine two well-known techniques to find the edges or boundaries in a video. We reduce the dimensionality of the data and put them all into the same format. The SVD and its application to color and word data were described in Section 8.2. We describe the test material we use to illustrate our algorithm in Section 8.3.3.

Scale-space techniques give us a way to analyze temporal regions of the video that span a time range from a few seconds to tens of minutes. Properties of scale spaces and their application to segmentation are described in Section 8.3.4.

In Section 8.3.5, we describe our algorithm, which combines these two approaches.

We discuss several temporal properties of video, and present simple segmentation results, in Section 8.3.6. Our representation of video allows us to measure and compare the temporal properties of the color and words. We perform a hierarchical segmentation of the video, automatically creating a table of contents for the video.

We conclude in Section 8.3.7 with some observations about this representation.

8.3.3 Test Material

We evaluated our algorithm using the transcript from two different videos.

The shortest test was the manual transcript of a 30 minute CNN Headline News television show [Linguistic Data Consortium, 1997]. This transcript is cleaner than those typically obtained from closed-captioned data or automatic speech recognition.

We also looked at the words and images from a longer documentary video, "21st Century Jet," about the making of the Boeing 777 airplane [PBS Home Video, 1995]. We analyzed the color information from the first hour of this video, and the words from all six hours.

In these two cases we have relatively clean transcripts and the ends of sentences are marked with periods. We can also use automatic speech recognition (ASR) to provide a transcript of the audio, but sentence boundaries are not reliably provided by ASR systems. In that case, we divide the text arbitrarily into 20 word groups or "sentences." We believe that a statistical technique such as LSI will fail gracefully in the event of word errors. For the remainder of this chapter we will use the word "sentence" to indicate a block of text, whether ended by a period or found by counting words.

8.3.4 Scale Space

Witkin [Witkin, 1984] introduced the idea of using scale-space segmentation to find the boundaries in a signal. In scale space, we analyze a signal with many different kernels that vary in the size of the temporal neighborhood that is included in the analysis at each point in time. If the original signal is $s(t)$, then the scale-space representation of this signal is given by

$$s_\sigma(t) = \int s(\tau) g(\sigma, t - \tau) d\tau, \tag{8.7}$$

where $g(\sigma, t - \tau)$ is a Gaussian kernel with a variance of σ. With σ approaching zero, $s_\sigma(t)$ is nearly equal to $s(t)$. For larger values of σ, the resulting signal, $s_\sigma(t)$, is smoother because the kernel is a low-pass filter. We have transformed a one-dimensional signal into a two-dimensional image that is a function of t and σ.

An important feature of scale space is that the resulting image is a continuous function of the scale parameter, σ. Because the location of a local maximum in scale space is well behaved [Babaud et al., 1986], we can start with a peak in the signal at the largest scale and trace it back to the exact point at zero scale where it originates. The range of scales over which the peak exists is a measure of how important this peak is to the signal.

In scale-space segmentation, we look for changes in the signal over time. We do so by calculating the derivative of the signal with respect to time and then finding the local maximum of this derivative. Because the derivative and the scale-space filter are linear, we can exchange their order. Thus, the properties of the local maximum described previously also apply to the signal's derivative.

Lyon [Lyon, 1984] extended the idea of scale-space segmentation to multi-dimensional signals, and used it to segment a speech signal. The basic idea remains the same: He filtered the signal using a Gaussian kernel with a range of scales. By performing the smoothing independently on each dimension, he traced with the new signal a smoother path through his 92-dimensional space. To segment the signal, he looked for the local peaks in the magnitude of the vector derivative.

Cepstral analysis transforms each vocal sound into a point in a high-dimensional space. This transformation makes it easy to recognize each sound (good for automatic speech recognition) and to perform low-level segmentation of the sound (as demonstrated by Lyon). Unfortunately, the cepstral coefficients contain little information about high-level structures. Thus, we consider the image and the semantic content of the video.

Combining LSI analysis with scale-space segmentation is straightforward. This process is illustrated in Figure 8.1. We describe the scale-space process as applied to semantic content. The analysis of the acoustic and color data is identical to the semantic information.

The semantic data is first grouped into a time sequence of sentences, s_i. From these groups, we create a histogram of word frequencies, $\vec{H}(s_i)$, a vector function of sentence number s_i. LSI/SVD analysis of the full histogram produces a k-dimensional representation, $\vec{H}_k(s_i) = X^k$ of the document's semantic path (where the dimensionality k is much less than the original histogram.) In this work[2] we arbitrarily set $k = 10$.

We use a low-pass filter on each dimension of the reduced histogram data $\vec{H}_k(s_i)$, replacing s in equation 8.7 with each component of $\vec{H}_k(s_i) =$

[2]Information retrieval systems often use 100–300 dimensions to distill thousands of documents, but those collections cover a larger number of topics than we see in a single document.

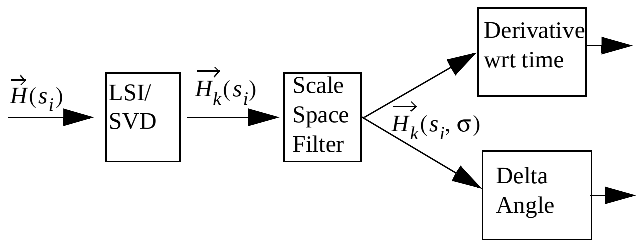

Figure 8.1. The LSI-SS algorithm. The top path shows the derivative based on euclidean distance. The bottom path shows the proper distance metric for LSI based on angle. See Section 8.3.4 for definitions.

$[H_1(s_i)H_2(s_i)...H_k(s_i)]^T$ to find a low-pass filtered version of the semantic path. This replacement gives $\vec{H}_k(s_i, \sigma)$, a k-dimensional vector function of sentence number and scale.

We are interested in detecting edges in acoustic, color and semantic scale spaces. An important property of the scale-space segmentation is that the length of a boundary in scale space is a measure of the importance of that boundary. It is useful to think about a point representing the document's local content wandering through the space in a pseudo-random walk. Each portion of the video is a slightly different point in space, and we are looking for large jumps in the topic space. As we increase the scale, thus lowering the cutoff frequency of a low-pass filter, the point moves more sluggishly. It eventually moves to a new topic, but small variations in topic do not move the point much. Thus, the boundaries that are left at the largest scales mark the biggest topic changes within the document.

The distance metric in Witkin's original scale-space work [Witkin, 1984] was based on Euclidean distance. When we use LSI as input to a scale-space analysis, our distance metric is based on angle. The dot product of adjacent (filtered and normalized) semantic points gives us the cosine of the angle between the two points. We convert this value into a distance metric by subtracting the cosine from one.

When we use LSI within a document, we must choose the appropriate block size. Placing the entire document into a single histogram gives us little information that we can use to segment the document. On the other hand, one-word chunks are too small; we would have no way to link single-word subdocuments. The power of LSI is available for segments that comprise a small chunk of text, where words that occur in close proximity are linked together by the histogram data.

Choosing the proper segment size is straightforward during the segmentation phase, since projecting onto a subspace is a linear operation.

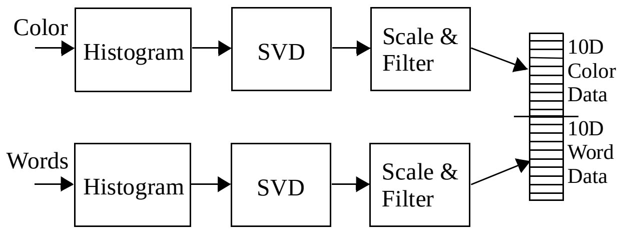

Figure 8.2. Combining color, words and scale space analysis. The result is a 20-dimensional vector function of time and scale.

Thus, even if we start with single-word histograms, the projection of the (weighted) sum of the histograms is the same as the (weighted) sum of the projections of the histograms.

The story is not so simple with the SVD calculation. For this study, we chose a single sentence as the basic unit of analysis, based on the fact that one sentence contains one subject. It is possible that larger subdocuments, or documents keyed by other parameters of a video, such as color information, might be more meaningful. The results of the temporal studies, described in Section 8.3.6.0, suggest that the optimal segment size is four to eight sentences, or a paragraph.

8.3.5 Combined Image and Audio Data

Our system for hierarchical segmentation of video combines the audio (semantic) and image (color) information into a single unified representation, and then uses scale-space segmentation on the combined signal (Figure 8.2).

Our algorithm starts by analyzing a video, using whatever audio and image features are available. For this chapter, we concentrated on the color and the semantic histograms. We perform an SVD on each feature, gaining noise tolerance and the ability to handle synonyms and polysemy (see Section 8.2.3).

The SVD, for either the color or the words, is performed in two steps. We build a model by collecting all the features of the signal into a matrix and then computing that matrix's SVD to find the k left-singular vectors that best span the feature space. We use the model by projecting the same data onto these k-best vectors to reduce the dimensionality of the signal. The semantic information typically starts with more than 1,000 dimensions; the color information has 512 dimensions. For the examples described in this chapter, we reduced all signals to individual 10-dimensional spaces.

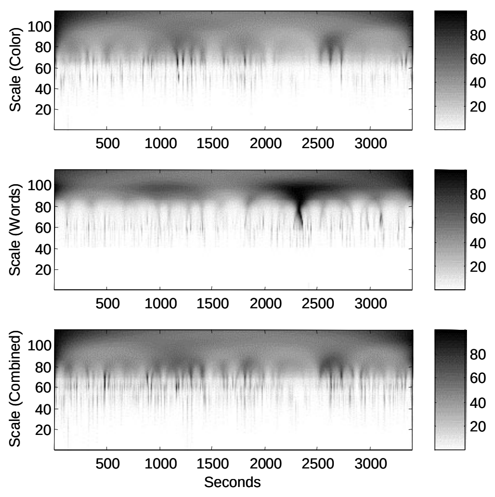

Figure 8.3. These three plots show the derivatives of the scale space representations for the colors (top), words (middle) and combined (bottom) spaces of the Boeing 777 video. Many details are lost because the 102089 frames are collapsed into only a few inches on this page.

The challenge when combining information in this manner is to not allow one source of information to overwhelm the others. The final steps before combining the independent signals are scaling and filtering. Scaling confers similar power on two independent sources of data. Typically, color histograms have larger values, since the number of pixels in an image tends to be much greater than the number of words in a semantic segment. Without scaling, the color signal is hundreds of times larger than the word signal; the combined signal makes large jumps at every color change, whereas semantic discontinuities have little effect.

To avoid this problem and to normalize the data, we balance the color and the semantic vectors such that both had an average vector magnitude of 1. Other choices are possible; for example, one might decide that the semantic signal contains more information about content changes than does the image signal and thus should have a larger magnitude. Plots showing the derivative of the color, word and the combined scale spaces are shown in Figure 8.3.

As we will discuss in Section 8.3.6.0, each signal has a natural frequency content, which we can filter to select a scale of interest. Thus, it might be appropriate to high-pass filter the color information to minimize the effects of changes over time scales greater than 10 seconds, while low-pass filtering the semantic information to preserve the information over scales greater than 10 seconds. We did not do this kind of filtering for the results presented in this chapter.

We combined the audio and visual data by aligning and concatenating the individual vectors. Alignment (and resampling) is important because the audio and image data have different natural sampling rates. Typically, the color data are available at the frame rate, 29.97 Hz, whereas the word information is available only at each sentence boundary—which occurred every 8 seconds, on average, in the Boeing 777 video that we studied.

We marked manually the start of each sentence in the video's audio channel. The marking was approximate, delineating the beginning of each sentence within a couple of seconds. We then created a new 10-dimensional vector by replicating each sentence's SVD-reduced representation at all the appropriate frame times. Then, based on the approximate sentence delineations, we smoothed the semantic vector with a 2-second rectangular averaging filter.

We concatenate the video and semantic vectors at each frame time, turning two 10-dimensional signals, sampled at 29.97 Hz, into a single 20-dimensional vector. We then can use these data as input to the scale-space algorithm.

8.3.6 Hierarchical Segmentation Results

We evaluated our approach with two studies. First, we studied the temporal properties of videos and text, by characterizing the temporal autocorrelation of the color and semantic information in news and documentary videos (Section 8.3.6.0). Second, to quantify the results of our segmentation algorithm, we performed scale-space hierarchical segmentation on two multimedia signals and compared the results to several types of segmentations (Section 8.3.6.0).

Temporal Results. There are many ways to characterize the temporal information in a signal. The autocorrelation analysis we describe in this section tells us the minimum and maximum interesting temporal scales for the audio and image data. This information is important in the design and characterization of a segmentation algorithm.

Autocorrelation. We investigated the temporal information in the signals by computing the autocorrelation of our representations:

$$R_{xx}(\tau) = \int_{-\infty}^{\infty} x'(t)x'(t+\tau)dt, \tag{8.8}$$

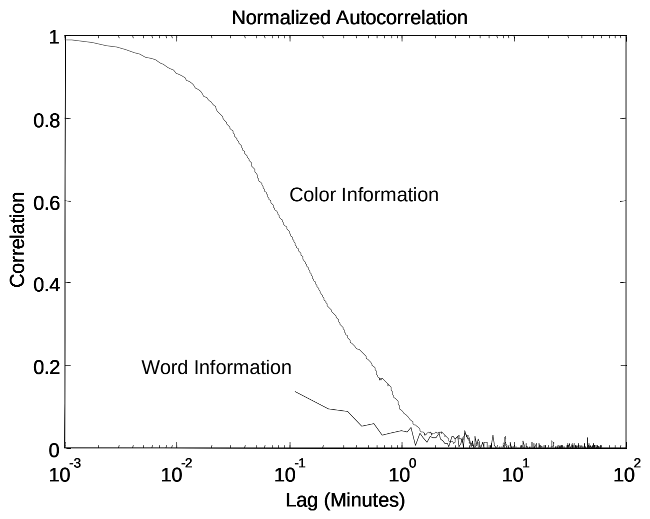

Figure 8.4. Color and word autocorrelations for the Boeing 777 video.

where x' is the original signal with the mean subtracted. There are six one-hour videos in the Boeing 777 documentary. The short length makes it difficult to estimate very long autocorrelation lags (more than 30 minutes). We computed the autocorrelation individually for each hour of video, then averaged the results across all videos to obtain a more robust estimate of the autocorrelation.

For both the image and the semantic data we used the reduced-dimensionality signals. We assumed that each dimension is independent and summed the autocorrelation over the first four dimensions to find the average correlation. The results of this analysis are shown in Figure 8.4 for both the image and the semantic signals.

The correlation for the color data is high until about 1/10 minute, when it falls rapidly to zero. This behavior makes sense, since the average shot length in this video, as computed by YesVideo (see Section 8.3.6.0), is 8 seconds.

Grouped Autocorrelation. At first, we were surprised by the semantic-signal results: There was little correlation at even the smallest time scale. We postulated that individual sentences have little in common with one another, but that groups of consecutive sentences might show more similarity. Usually, the same words are not repeated from one sentence to the next, and neighboring sentences should be nearly orthogonal.

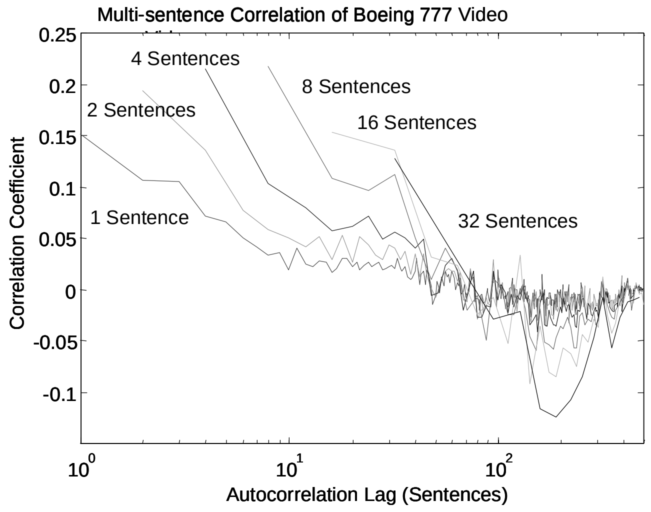

Figure 8.5. Grouping 4–8 sentences produces a larger semantic autocorrelation (data from the Boeing 777 video). This peak corresponds to 29–57 seconds of the original video.

By grouping sentences—averaging several points in semantic space—we formed a more robust estimate of the exact location of a given portion of a transcript or document in semantic space. In Figure 8.5, we show the results that we obtained by grouping sentences of the Boeing 777 video. In the line marked "8 sentences," we grouped (averaged) the reduced-dimensionality representation of eight contiguous sentences, and computed the correlations between that group and other groups of eight, non overlapping, sentences.

Figure 8.5 shows that, indeed, the correlation starts small when we consider individual sentences, and gradually grows to a maximum for groups of between four and eight sentences, and then falls again as the group size increases. Evidently, grouping four to eight sentences allows us to estimate reliably a single point in semantic space. The correlation reaches a minimum at approximately 200 sentences.

Interestingly, in two documents we saw a strong anti-correlation around 200 sentences [Slaney et al., 2001]. This is interesting because it indicates that the topic has moved from one side of the semantic space to the opposite side in the course of 200 sentences.

Segmentation Results. We evaluated our hierarchical representation's ability to segment the 30-minute Headline News television show and the first hour of the Boeing 777 documentary. We describe qualitative results and a quantitative metric, and show how our results compare to those obtained with automatic shot-boundary and manual topical segmentations.

Most videos are not organized in a perfect hierarchy. In text, the introduction often presents a number of ideas, which are then explored in subsequent sections; a graceful transition is used between ideas. The lack of hierarchy is much more apparent in a news show, the structure of which may be somewhat hierarchical, but is designed to be watched in a linear fashion. For example, the viewer is teased with information about an upcoming weather segment, and the "top of the news" is repeated at various stages through the broadcast.

We illustrate our hierarchical segmentation algorithm by showing intermediate results using just the semantic information from the Headline News video. The results of hierarchical segmentations are compared with the ground truth. The LDC [Linguistic Data Consortium, 1997] provided story boundaries for this video, but we estimated the high-level structure based on our familiarity with this news program. The timing and other meta information were removed from the transcript before analysis. We found 257 sentences in this broadcast transcript; which after the removal of stop words, contained 1032 distinct words.

Intermediate Results. Our segmentation algorithm measured the changes in a signal over time as a function of the scale size. A scale-space segmentation algorithm produced a boundary map showing the edges in the signal, as shown in Figure 8.6. At the smallest scale there were many possible boundaries; at the largest scale, with a long smoothing window, only a small number of edges remained.

Due to the local peculiarities of the data, the boundary deviated from its true location as we moved to large windows. We traced the boundary back to its true location (at zero scale) and drew the straightened boundary map shown at the bottom of Figure 8.6. For any one boundary, indicated by its vertical lines, strength is represented by line height, and is a measure of how significant this topic change is to the document.

Qualitative Measure. The classic measures for the evaluation of text-retrieval performance [Allan et al., 1998] do not extend easily to a system that has hierarchical structure. Instead, we evaluated our results by examining a plot that compared headings and the scale-space segmentation strength. The scale-space analysis produced a large number of possible segmentations; for each study, we plotted only twice the number of boundaries indicated by the ground truth.

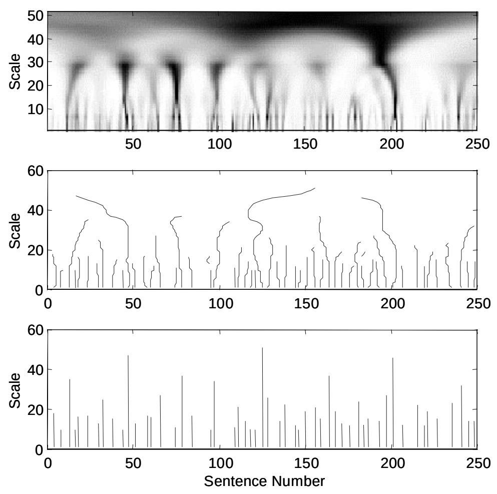

Figure 8.6. Representations of the semantic information in the Headline News video in scale space. The top image shows the cosine of the angular change of the semantic trajectory with different amounts of low-pass filtering. The middle plot shows the peaks of the scale-space derivative for the tomography chapter. The bottom plot shows the peaks traced back to their original starting point. These peaks represent topic boundaries.

Our results of calculating the hierarchical segmentations of the Headline News are shown in Figure 8.7. On the right, the major (left most text) and the minor (right most text) headings are shown. The left side of the plot shows the strength of the boundary. The "Weather," "Tech Trends" and "Lifestyles" sections are indicated within a few sentences, yet there are large peaks at other locations in the transcript. Interestingly, there is a large boundary near sentence 46, which neatly divides the softer news stories at the start of this broadcast from the political stories that follow.

We measured the degree of agreement between two segmentations by looking at both ends of a fixed window passed over two sets of segmentation data. Figure 8.8 summarizes the process for one set of data labeled with ground truth and for another set labeled "experimental." The segmentation, at this point, is successful if both ends of the window fall within the same segment or if each is in a different segment. The segmentation is wrong if, for example, the ground-truth window falls entirely within one segment and the experimental window covers two or more segmentation boundaries. We move the window over the entire document and calculate the fraction of correct windows.

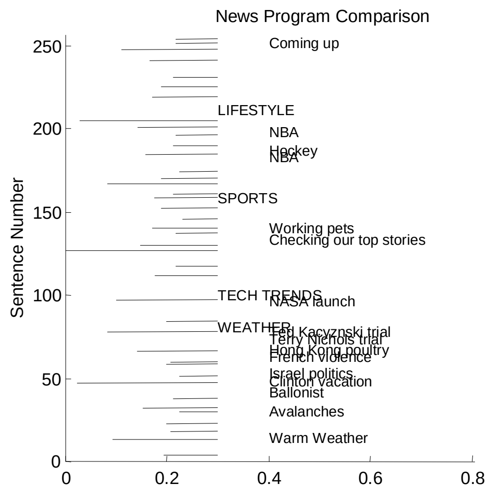

Figure 8.7. A comparison of ground truth (right) and the size of boundaries for the Headline News video as determined by scale-space segmentation. The major headings are in all capitals, and the sub-headings are in upper and lower case.

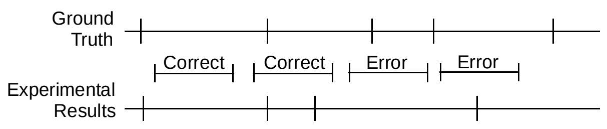

Figure 8.8. We evaluate accuracy by measuring whether the ends of a fixed-size window fall in the same or different segments.

Our quantitative measure of segmentation performance was suggested by Lafferty and refined by Doddington [Allan et al., 1998] An especially important property of Lafferty's measure for semantic segmentations is that small offsets in the segmentation lower the performance metric, but do not cause complete misses. We used a fixed window size that was 50 percent of the length of the average segment calculated using the ground-truth segmentation.

Lafferty's segmentation metric has several properties that were reflected in our data. Assume that the probability that any particular frame is a boundary is independent and is fixed at $p = 1/(2N)$ where N is the window length or half the average segment length. If we measure the accuracy of an experimental result with just one (large) segment, then Lafferty's measure is asymptotically equal to

$$(1-p)^N \approx 0.606. \tag{8.9}$$

Conversely, if we measure the accuracy of a segmentation that puts a boundary at every time step, then Lafferty's measure is equal to

$$1-(1-p)^N \approx 0.394. \tag{8.10}$$

Finally, if we compare two random segmentations, each with the same probability of a boundary, Lafferty's measure indicates that the segmentation accuracy is

$$(1-p)^{2N} + \left[1-(1-P)^N\right]^2 \approx 0.523. \tag{8.11}$$

Shot Boundary Segmentation. We used the segmentation produced by a state-of-the-art commercial product, designed by YesVideo [YesVideo, Inc., 2002], as our shot-boundary ground truth. They reported that, on a database of professionally produced wedding videos, their segmenter had an overall precision of 93% and a recall of 91%. For their test set, most of the errors, both false positives and false negatives, were due to uncompensated camera motion.

We performed quantitative tests on the first hour of the Boeing 777 video. This video had 102,089 frames. The semantic analysis found 1314 distinct words in 537 sentences. There were shot boundaries on average every 242 frames. The standard deviation of the Gaussian blur used in scale-space filtering is $\sigma = 1.1^{s-1}$, where s is the scale number.

To evaluate our combined representation, we show the results here using only the color data, only the word data, and the combination. We evaluate Lafferty's measure for the segmentation boundaries predicted at each scale, effectively assuming that a single scale would produce the best segmentation. Assuming all segmentation boundaries are at the

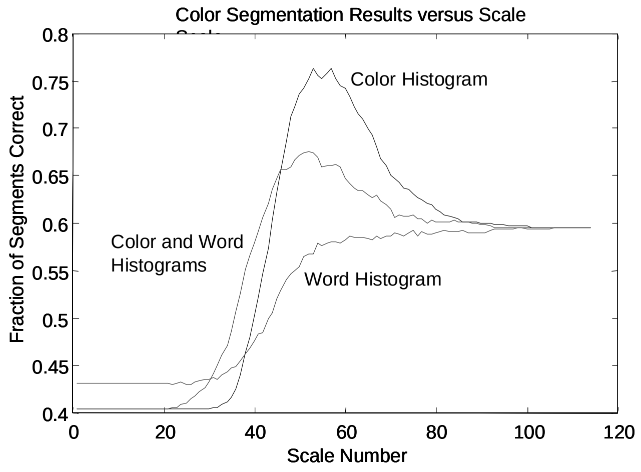

Figure 8.9. This figure shows the accuracy of the scale-space segmentation algorithm, at any one scale, at finding shot boundaries. Video was the first hour of the Boeing 777 video, compared to ground truth from YesVideos segmenter [YesVideo, Inc., 2002].

same scale is not the best solution; instead, the information from the scale-space segmentation metric should be used as input to a higher-level model of video transitions, as suggested by Srinivasan [Srinivasan et al., 1999].

Figure 8.9 shows how segmentation accuracy varied with scale, comparing the segmentation at each scale to the YesVideo results. At small scales the probability, as predicted by equation 8.10, was 40%. At large scales, only one or two boundaries were found, and, as predicted by equation 8.9, the accuracy was 60%. At the middle scale, the segmentation accuracy was 77%—well above that of random segmentations (52%). As expected, the semantic signal does not predict the color boundaries. Adding the semantic information to the color information does reduce the highest accuracy at any one scale to 67%.

Semantic Segmentation. We also compared our algorithm's semantic segmentations to those of humans. Two of the authors of this chapter and a colleague segmented the transcript of the first hour of the Boeing 777 video. There was a wide range in what these three readers described as a segment; they chose to segment the text with 29, 37, and 122 segment boundaries. They found it difficult to produce a hierarchical segmentation of the text. The video was designed to be watched in one sitting; it transitions smoothly from topic to topic, weaving a single story.

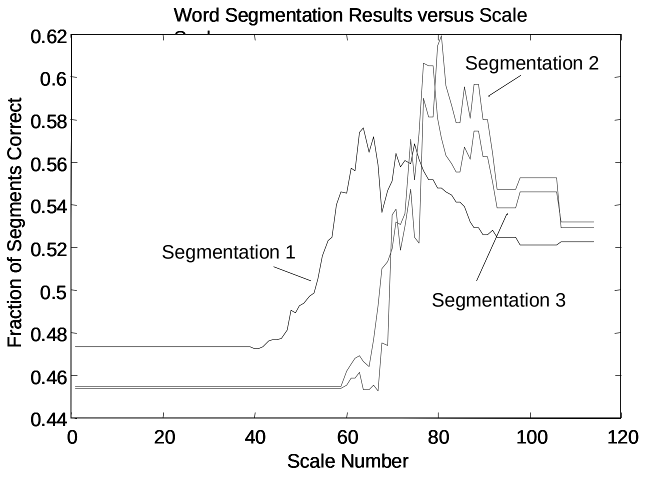

Figure 8.10. Manual segmentation versus scale, tested with Lafferty's measure (all three manual segmentations). Source data from the first hour of the Boeing 777 video.

Figure 8.10 shows how the scale-space segmentation compares, across scale, to the manual segmentations. As expected, the best scale is larger than that shown for the color segmentations in Figure 8.9. The scale-space segmentation algorithm matches each of the humans' segmentations equally well. Perhaps most surprisingly, the color information is a good predictor of the semantic boundaries. This correlation may indicate that the color signal carries information regarding content changes that is richer than we assumed.

8.3.7 Segmentation Conclusions

We have demonstrated a new framework for combining into a unified representation and for segmenting information from multiple types of information from a video. We used the SVD to reduce the dimensionality of each signal. Then, we applied scale-space segmentation to find edges in the signals that corresponded to large changes. We demonstrated how these ideas apply to words and to color information from a video.

These techniques are an important piece of a complete system. The system we have described does not have the domain knowledge to know that, for example when it is considering a videotape of a news broadcast, the phrase "coming up after the break" is a pointer to a future story and is not a new story in its own right. Systems that include domain knowledge about specific types of video content [Dharanipragada et al., 2000] show how this knowledge is incorporated.

We have described the natural-frequency content of information in a video. Autocorrelation analysis showed that the color information was correlated for about 0.1 minute, whereas the semantic content showed significant correlation for hundreds of sentences (tens of minutes). These results suggest the smallest meaningful unit of semantic information is about 8 sentences.

We described our hierarchical segmentation results by comparing them to conventional segmentations. Qualitatively, the automatic segmentation has many similarities to a manual segmentation. It is hard to evaluate the quantitative results, but we were surprised by the amount of information that was available in the color information for topical segmentation.

The methods we described here are equally useful with other information from a video. This includes speaker identification features, musical key, speech/music indicators, and even audio emotion. These techniques do not give any assistance with professional video production techniques, such as L-cuts, which change the audio topic and the camera shot at different times.

8.4 Semantic Retrieval

The previous section described an algorithm which used the semantic (and perhaps also the color information) to segment media. In this section, we build links between the media and the semantic content.

This section describes a method of connecting sounds to words, and words to sounds. Given a description of a sound, the system finds the audio signals that best fit the words. Thus, a user might make a request with the description "the sound of a galloping horse," and the system responds by presenting recordings of a horse running on different surfaces, and possibly of musical pieces that sound like a horse galloping. Conversely, given a sound recording, the system describes the sound or the environment in which the recording was made. Thus, given a recording made outdoors, the system says confidently that the recording was made at a horse farm where several dogs reside.

A system that has these functions, called MPESAR (mixtures of probability experts for semantic–audio retrieval), learns the connections between a semantic space and an acoustic space. Semantic space maps words into a high-dimensional probabilistic space. Acoustic space describes sounds by a multidimensional vector. In general, the connection between these two spaces will be many to many. Horse sounds, for example, might include footsteps and neighs.

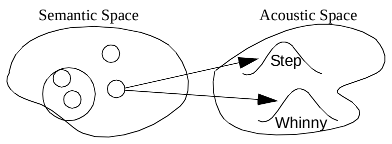

Figure 8.11. MPESAR models all of semantic space with overlapping multinomial clusters, each portion in the semantic model is linked to equivalent sound documents in acoustic space with a GMM.

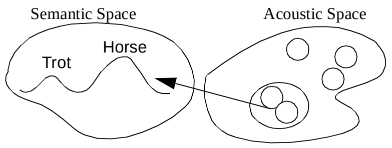

Figure 8.12. MPESAR describes with words an audio query by partitioning the audio space with a set of acoustic models and then linking each cluster of audio files (or documents) to a probability model in semantic space.

Figure 8.11 shows one half of MPESAR: how to retrieve sounds from words. Annotations that describe sounds are clustered and represented with multinomial models. The sound files, or acoustic documents, that correspond to each node in the semantic space are modeled with Gaussian mixture models (GMMs). Given a semantic request, MPESAR identifies the portion of the semantic space that best fits the request, and then measures the likelihood that each sound in the database fits the GMM linked to this portion of the semantic space. The most likely sounds are returned to satisfy the user's semantic request.

Figure 8.12 shows the other half of MPESAR: how to generate words to describe a sound. MPESAR analyzes the collection of sounds and builds models for arbitrary sounds. This approach gives us a multidimensional representation of any sound, and a distance metric that permits agglomerative clustering in the acoustic space. Given an acoustic request, MPESAR identifies the portion of the acoustic space that best fits the request. Each portion of the acoustic space has an associated multinomial word model, and from this model MPESAR generates words to describe the query sound.

In general, sounds that are close in acoustic space might correspond to many different points in semantic space, and vice versa. Thus, MPESAR builds two completely separate sets of models: one connecting audio to semantic space and the other connecting semantic to audio space.

8.4.1 The Algorithm

Mixture of Probability Experts. MPESAR uses a mixture of experts approach [Waterhouse, 1997] to link semantic and audio spaces. A mixture of experts approach uses a different expert for different regions of an input space. Thus, one expert might be responsible for horse sounds while another is responsible for bird sounds.

Mathematically, a mixture of probability experts for semantic to audio retrieval is summarized by the following equation

$$P(a|q) = \sum_{c} P(c|q)P(a|c) \tag{8.12}$$

Here $P(c|q)$ represents the probability that a semantic query (q) matches a cluster (c). The probability that a particular portion of acoustic space is associated with an expert or cluster (c) is given by $P(a|c)$. To find the overall probability of a point in audio space given the query, $P(a|q)$, we sum over all possible clusters, essentially interpolating the different expert's opinions to arrive at the final probability estimate.

We want to calculate the probability of a cluster given a query. We group semantic documents into clusters and then estimate $P(q|c)$. Using Bayes' rule: $P(c|q) = P(c)P(q|c)/P(q)$. The $P(c)$ and $P(q|c)$ terms are calculated using clustering algorithms described in Sections 8.4.1.0 and 8.4.1.0 Since the query is given, we can ignore the $P(q)$ term. The same formalism is used for the audio to semantic problem.

Semantic Features. MPESAR uses multinomial models to represent and cluster a collection of semantic documents. The likelihood that a document matches a given multinomial model is described by $L = \prod p_i^{n_i}$, where p_i is the probability that word i occurs in this type of document, and n_i is the number of times that word i is found in this document. The set of probabilities, p_i, is different for different types of documents. Thus, a model for documents about cows will have a relatively high probability for containing "cow" and "moo," whereas a model for documents that describe birds with have a high probability of containing "feather." These multinomial models accomplish the same task as LSI—convert a bag of words into a multinomial vector—but have a more principled theory.

A semantic document contains the text used to describe an audio clip. MPESAR uses the PORTER stemmer to remove common suffixes from the words, and deletes common words on the SMART list before further processing [Porter, 1980]. In effect, a 705-dimensional vector (the multinomial coefficients) describes a point in semantic space, and MPESAR partitions the space into overlapping clusters of regions.

Smoothing is used in statistical language modeling to compensate for a paucity of data. It is called smoothing because the probability associated with likely events is reduced and distributed to events that were not seen in the training data. The most successful methods [Chen and Goodman, 1996] use a back-off method, where data from simpler language models are used to set the probability of rare events. MPESAR uses a unigram word model, so the back-off model suggests a uniform low probability for all words.

Acoustic Features. Sound is difficult to analyze because it is dynamic. The sound of a horse galloping is constantly changing at time scales in the hundreds of milliseconds; a hoofstep is followed by silence, and then by another hoofstep. Yet we would like a means to transform the sound of a galloping horse into a single point in an acoustic space. This section describes acoustic features that allow us to describe each sound as a single point in acoustic space, and to cluster related sounds.

Conventional acoustic features for speech recognition and for sound identification use a short-term spectral slice to characterize the sound at 10-ms intervals. A combination of signal-processing and machine-learning calculations endeavors to capture the sound of a horse as a point in auditory space.

MFCC (mel-frequency cepstral coefficient) is a popular technique in the automatic speech recognition community to analyze speech sounds. Based on auditory perception, the MFCC representation captures the overall spectral shape of a sound, while throwing away the pitch information. This allows MFCC to distinguish one vowel from another or one instrument from another; while ignoring the melody. We use MFCC to reduce an audio signal from it's original representation (22kHz sampling rate) to a 13-dimensional vector sampled 100 times a second.

MFCC has been used in earlier music segmentation work [Foote, 1999], but it is not the ideal representation. In this work it allows us to capture the overall timbral qualities of the sound, but ignore the detailed pitch information. The dimensionality reduction is similar to that performed by an SVD, but takes into account specialized knowledge about audio and dimensions that are safe to ignore. The MFCC representation does not give us any high-level information about rhythm or other musical properties of the signal. Eventually we hope that more sophisticated acoustic features will allow us to segment musical accompaniment at phrase or beat boundaries, or even to detect mood changes in movie scores.

The process of converting a waveform into a point in acoustic space is shown in Figure 8.13. The MFCC algorithm [Quatieri, 2002] decomposes each signal into broad spectral channels and compresses the loudness of the signal. RASTA filtering [Quatieri, 2002] is used on the MFCC coefficients to remove long-term spectral characteristics that often occur due to the different recording environments. Then seven frames of data—

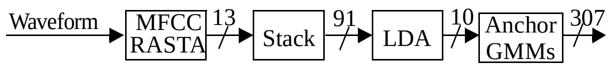

Figure 8.13. The acoustic signal processing chain. Arrows are marked with the signals dimensionality. All but the last are sampled at 100Hz. The final output is sampled once per sound.

three before the current frame, the current frame, and the three frames following the current frame—are stacked together. Finally, linear discriminant analysis (LDA) [Quatieri, 2002] uses the intra- and inter-class scatter matrices for a hand-labeled set of classes to project the data onto the optimum dimensions for linear separability.

The long-term temporal characteristics of each sound are captured using a GMM. One of the Gaussians might capture the start of the footstep, a second captures the steady-state portion, a third captures the footstep's decay, and, finally, a fourth captures the silence between footsteps. The GMM measures the probability that a vector sequence fits a probabilistic model learned from the training sounds. Unlike hidden Markov models (HMMs), a GMM ignores temporal order.

MPESAR converts the MFCC-RASTA-LDA plus GMM recognition system into an auditory space by using model likelihood scores to measure the closeness of a sound to pre-trained acoustic models. The negative log-likelihood that a sound fits a model is a measure of the distance of the new sound from the test model.

Acoustic to Semantic Lookup. Given representations of acoustic and semantic spaces, we can now build models to link the two spaces together. The overall algorithm for both acoustic to semantic and semantic to acoustic lookup is shown in Figure 8.14.

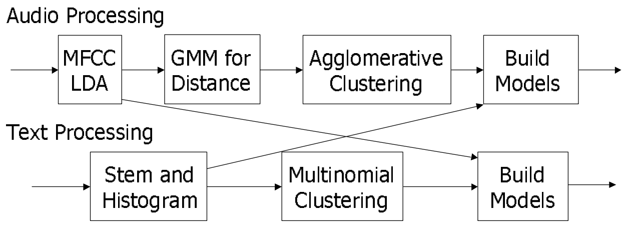

Figure 8.14. A schematic showing the process of building the MPESAR models. The top line shows the construction of the audio to semantic model and the bottom line shows the construction of the semantic to audio model.

Acoustic space is clustered into regions using agglomerative clustering [Jain and Dubes, 1988]. We compute the distance between each pair of training sounds $[L(model\ a|sound\ b) + L(model\ b|sound\ a)]/2$ where $L(modela|soundb)$ represents the likelihood that sound b is generated by model a. At each step, agglomerative clustering grows another layer of a hierarchical model by merging the two remaining clusters that have between them the smallest distance. MPESAR uses "complete" linkage, which uses the maximum distance between the points that form the two clusters, to decide which clusters should be combined. While agglomerative clustering generates a hierarchy, MPESAR only uses the information about which sounds are clustered. Leaves at the bottom of the tree are considered clusters containing a single document.

Each acoustic cluster is composed of a number of audio tracks and their associated descriptive text. A new 10-element GMM with diagonal covariance models all the sounds in this cluster and estimates the probability density for acoustic frames in this cluster, $P(a|c)$. Given a new sound, MPESAR uses this model to estimate the probability that a new sound belongs to this cluster. The text associated with each acoustic sample in the cluster is used to estimate the semantic model associated with this cluster. This is written as a simple multinomial model; there is not enough text in this study to form a richer model.

Given a new waveform, MPESAR queries all acoustic GMMs to find the probability that each possible cluster generated this query. Each cluster comes with an associated semantic model. MPESAR uses a weighted average of all the semantic models, based on cluster probabilities, to estimate the semantic model that describes the test sound. The words that describe the test sound are entries in the semantic multinomial model with the highest probabilities.

Semantic to Acoustic Lookup. A similar procedure is used for semantic to acoustic lookup. A document's point in semantic space is described by the coefficients of a unigram multinomial model. Semantic space is clustered into regions using a multinomial clustering algorithm, which uses an iterative expectation-maximization algorithm [Nigam et al., 1998] to group documents with similar (multidimensional) models. In this work, we assign each document to its own cluster, and then split the entire corpus into a number of arbitrary-sized clusters (32, 64, 128 and 256 clusters for the corpus).

Each text cluster is composed of a number of text documents and their associated audio tracks. All the text associated with each cluster is used to form a unigram multinomial model of the text documents. All of the audio associated with a cluster is used to form a 10-element

GMM to describe the link to audio space. (Note there are three sets of GMMs used in this work: the GMMs used to compute the distances as part of audio clustering, the GMMs used to model each audio cluster, and the GMMs used to model the sounds associated with each semantic cluster.)

Given a text query, MPESAR finds the probability that each semantic cluster generated the query. Then the acoustic models are averaged (weighted by the cluster probabilities) to find the probability that any one sound fits the query.

8.4.2 Testing

This section describes several tests performed using the algorithms described above.

Data. The animal sounds from two sets of sound effect CDs were used as training and testing material. Seven CDs from the BBC Sound Effects Library (#6, 12, 30, 34, 35, 37, 38) contained 261 separate tracks and 390 minutes of animal sounds. Two CDs from the General 6000 Sound Effect library (all tracks from CD6003 and tracks 18 to 40 of CD6023) totaled 122 tracks and 110 minutes of animal sounds.

The concatenated name of the CD (e.g., "Horses I") and track description (e.g., "One horse eating hay and moving around") forms a semantic label for each track. The audio from the CD track and the liner notes form a pair of acoustic and semantic documents used to train the MPESAR system.

The system training and testing described in this chapter were performed on distinct sets of data. 80% of the tracks (307) from both sets of CDs were randomly assigned as training data in the procedure shown in Figure 8.15. The remaining 20% of the tracks (93) were reserved for testing. Mixing the data obtained from the two sets of CDs is important for several reasons. First, the acoustic environments of the two data sets are different; RASTA reduces these effects. Second, the words and description are different because the sounds are labeled by different organizations with different needs. For example, the BBC describes the sound of a cat's vocalization as miaow and the General Sound Effects CD uses meow. Finally, the two sets of audio data do not contain the same sounds: There are many sounds in the General set which are not represented in the BBC training set.

Acoustic Feature Reduction and Language Smoothing. The audio-feature reduction using LDA was computed using portions of the audio data from both sets of CDs. We chose ten broad classes of dis-

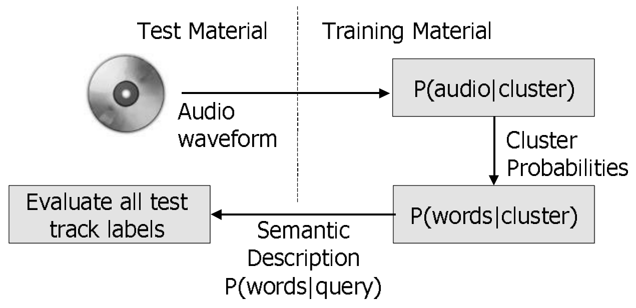

Figure 8.15. A schematic of the audio to semantic testing procedure.

tinct sound types (baboon, bird, cat, cattle, dog, fowl, goat, horse, lion, pig, sheep). The stacked features from only those audio tracks that fit these classes were used as input to the LDA algorithm. This computation produced a matrix that reduced the 91-dimensional data to the 10-dimensional subspace that best discriminates between these 10 classes. This dimensionality reduction was fixed for all experiments.

A simple test was used to set the amount of smoothing in the language models. Without smoothing, the semantic lookup results were poor because many of the General sounds were labeled with the word "animal," which was seldom used in the BBC labels. The results here were generated using a back-off method that added a small constant probability ($1/N_w$, where N_w is the number of words in the vocabulary) to each word model.

Labeling Tests. Figure 8.15 shows the test procedure for the acoustic-to-semantic task (a similar procedure is used to test semantic-to-audio labeling.) Audio from each test track is applied as an acoustic query to the system. The MPESAR system calculates the probability of each cluster given this acoustic query. These cluster probabilities are used to weight the semantic models associated with each cluster. The result is a multinomial probability distribution that represents the probabilities that each word in the dictionary describes the acoustic test track. The likelihoods that each test-track description fit the query's semantic description were sorted and the rank of the true test label was recorded.

Figures 8.16 and 8.17 show histograms of the true test ranks for both directions of the MPESAR algorithm. Figure 8.16 shows the acoustic-to-semantic results and the median rank of the true result over all the test tracks is 17.5. Figure 8.17 shows the semantic-acoustic results and the median rank of the true result for this direction is 9. At this point

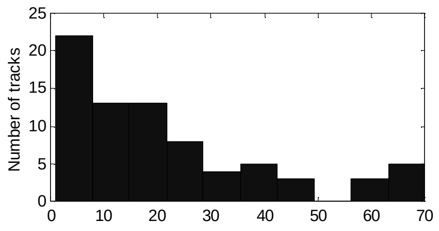

Figure 8.16. Histogram of true label ranks based on likelihoods from audio-to-semantic tests.

we do not understand the difference in performance between these two directions.

8.4.3 Retrieval Conclusions

This chapter described a system that uses a mixture of probability experts to learn the connection between an audio and a semantic space, and the reverse. It describes the conversion of sound and text into acoustic and semantic spaces and the process of creating the mixture of probability experts. The system was tested using commercial sound-effect CDs and is effective at labeling acoustic queries with the most appropriate words, and for finding sounds that fit a semantic query.

There are several improvements to this system that are worth pursuing. First, an algorithm that integrates the clustering and the MPE training will improve the system's models. Second, a richer acoustic description, perhaps replacing the GMMs with hidden Markov models, will provide more discrimination power. Finally, larger training sets will improve the system's knowledge.

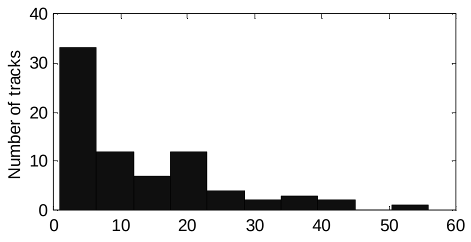

Figure 8.17. Histogram of true label ranks based on likelihoods from semantic-to-audio tests.

8.5 Acknowledgements

We appreciate the assistance that we received from Byron Dom, Arnon Amir, Myron Flickner, John Fisher, Clemens Drews and Michele Covell. Ian Nabney's NETLAB software was used to calculate the GMMs; Roger Jang provided the clustering code.

References

Allan, J., Carbonell, J., Doddington, G., Yamron, J., and Yang, Y. (1998). Topic detection and tracking pilot study: Final report.

Aner, A. and Kender, J. R. (2002). Video summaries through mosaic-based shot and scene clustering. In *ECCV*, pages 388–402.

Babaud, J., Witkin, A. P., Baudin, M., and Duda, R. O. (1986). Uniqueness of the gaussian kernel for scale-space filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):26–33.

Barnard, K. and Forsyth, D. (2001). Learning the semantics of words and pictures. In *Proceedings of the 2001 International Conference on Computer Vision*, volume 2, pages 408–415.

Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In Joshi, A. and Palmer, M., editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 310–318, San Francisco. Morgan Kaufmann Publishers.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

Dharanipragada, S., Franz, M., McCarley, J. S., Papineni, K., S.Roukos, T.Ward, and Zhu, W.-J. (2000). Statistical models for topic segmentation. In *Proc. of ICSLP-2000.*

Dumais, S. T. (1991). Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23:229–236.

Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D., Steele, D., and Yanker, P. (1993). The qbic project: Query images by content using color, texture and shape. In *SPIE Storage and Retrieval of Image and Video Database*, pages 173–181.

Foote, J. (1999). Visualizing music and audio using self-similarity. In *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 77–80. ACM Press.

Hearst, M. (1994). Multi-paragraph segmentation of expository text. In *32nd. Annual Meeting of the Association for Computational Linguis-*

tics, pages 9–16, New Mexico State University, Las Cruces, New Mexico.

Jain, A. K. and Dubes, R. C. (1988). *Algorithms for Clustering Data.* Prentice-Hall.

Leung, Y., Zhang, J. S., and Xu, Z. B. (2000). Clustering by scale-space filtering. In *IEEE Transactions on PAMI*, volume Vol. 22(12), pages 1396–1410.

Linguistic Data Consortium (1997). 1997 english broadcast news speech (hub-4).

Lyon, R. F. (1984). Speech recognition in scale space. In *Proc. of 1984 ICASSP*, pages 29.3.1–4.

Miller, N. E., Wong, P. C., Brewster, M., and Foote, H. (1998). TOPIC ISLANDS - A wavelet-based text visualization system. In Ebert, D., Hagen, H., and Rushmeier, H., editors, *IEEE Visualization '98*, pages 189–196.

Nigam, K., McCallum, A. K., Thrun, S., and Mitchell, T. M. (1998). Learning to classify text from labeled and unlabeled documents. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, pages 792–799, Madison, US. AAAI Press, Menlo Park, US.

PBS Home Video (1995). 21st century jet: The building of the 777.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.

Quatieri, T. F. (2002). *Discrete-Time Speech Signal Processing: Principles and Practice.* Prentice-Hall.

Slaney, M. (2002). Mixtures of probability experts for audio retrieval and indexing. In *Proceedings 2002 IEEE International Conference on Multimedia and Expo*, volume 1, pages 345–348.

Slaney, M., Ponceleon, D., and Kaufman, J. (2001). Multimedia edges: finding hierarchy in all dimensions. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 29–40. ACM Press.

Srinivasan, S., Ponceleona, D., Amir, A., and Petkovic, D. (1999). 'what is in that video anyway?' in search of better browsing. In *Proceedings IEEE International Conference on Multimedia Computing and Systems*, pages 388–393.

Waterhouse, S. (1997). Classification and regression using mixtures of experts.

Witkin, A. P. (1984). Scale-space filtering: A new approach to multi-scale description. In *Proceedings of ICASSP*, pages 39A.1.1–39A.1.4.

YesVideo, Inc. (2002).