UNSUPERVISED CLUSTERING OF AMBULATORY AUDIO AND VIDEO

Brian Clarkson and Alex Pentland

Perceptual Computing
MIT Media Lab
Cambridge, MA 02139
{clarkson,sandy}@media.mit.edu

ABSTRACT

A truly personal and reactive computer system should have access to the same information as its user, including the ambient sights and sounds. To this end, we have developed a system for extracting events and scenes from natural audio/visual input. We find our system can (without any prior labeling of data) cluster the audio/visual data into events, such as passing through doors and crossing the street. Also, we hierarchically cluster these events into scenes and get clusters that correlate with visiting the supermarket, or walking down a busy street.

1. INTRODUCTION

Computers have evolved into miniature and wearable systems[7]. As a result there is a desire for these computers to be tightly coupled with their user's day-to-day activities. A popular analogy for this integration equates the wearable computer to an intelligent observer and assistant for its user. To fill this role effectively, the wearable computer needs to live in the same sensory world as its human user. [8]

Thus, a system is required that can take this natural and personal audio/video and find the coherent segments, the points of major activity, and recurring events. The field of multimedia indexing has wrestled with many of the problems that such a system creates. However, audio/video data that these researchers typically tackle are very heterogeneous and thus have little structure and what structure they do have is usually artificial, like scene cuts, and patterns in camera angle. The "eyes and ears" audio/video data that we are tackling is much more homogeneous and thus richer in structure, and filled with repeating elements and slowly varying trends.

The use of our system's resulting indexing differs greatly from the typical "querying for key-frames". Suppose our system has clustered its audio/video history into 100 models. Upon further use, the system notices that whenever the user requests his grocery list, model 42 is active. We would say that model 42 is the supermarket. However, the system does not need to have such a human-readable for model 42. (What would the system do with it? The user presumably knows already that he is in the supermarket.) However, a software agent built on our system would know to automatically display the user's grocery list when model 42 activates.

2. THE PERSONAL AUDIO-VISUAL TASK

In contrast to subject-oriented video or audio [5], such as TV [4], movies, and video recordings of meetings [3], our goal is to use video to monitor an individual's environment. Literally, the camera and microphone become an extra set of senses for the user. [1, 2]

2.1. Data Collection

In order to adequately sample the visual and aural environment of a mobile person, the sensors should be small and have a wide field of reception. The environmental audio was collected with a lavalier microphone (the size of a pencil eraser) mounted on the shoulder and directed away from the user. The environmental video was collected with a miniature CCD camera (1/4" diameter, 2" long) attached to a backpack (pointing backwards). The camera was fitted with a 180° wide-angle lens giving an excellent view of the sky, ground, and horizon at all times.

The system was worn around the city for a few hours, while the wearer performed typical actions, such as shopping for groceries, renting a video, going home, and meeting and talking with acquaintances. The resulting recording covered early to late afternoon (no night-time data). The camera's automatic gain control was used to prevent saturation in daylight.

2.2. Feature Extraction

Unlike the typical features used for face and speech recognition, we require features that are much less sensitive. We want our features to respond only to the most blindingly obvious events – walking into a building, crossing the street, riding an elevator. Since, our system is restricted to unsupervised learning, it is necessary to use robust features that do not behave wildly or respond to every change in the environment – only enough to convey the ambiance.

Video First the (r, g, b) pixel values were separated into (pseudo) luminance and chrominance channels:

$$I = r + g + b \quad I_r = r/I \quad I_g = g/I$$

The visual field of the camera was divided into 9 regions that correspond strongly to direction. The following features were extracted from the (I, I_r, I_g) values of each region:

$$\begin{bmatrix} \bar{I} & \bar{I_r} & \bar{I_g} \end{bmatrix}$$

$$Cov \begin{bmatrix} I^2 & II_r & II_g \\ & I_r^2 & I_r I_g \\ & & I_g^2 \end{bmatrix}$$

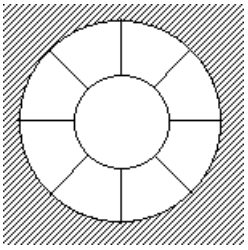

Figure 1: The 9 features on the left were extracted from each of the 9 regions shown on the right.

Hence, we are collapsing each region to a Gaussian in color space. This rough approximation lends robustness to small changes in the visual field, such as distant moving objects and small amplitude camera movement (the human body is not a stable camera platform).

Audio Auditory features were extracted with 25 Mel-scaled filter banks. The triangle filters give the same robustness to small variations in frequency (especially high frequencies), not to mention warping frequencies to a more perceptually meaningful scale.

Both the video and the audio features were calculated at a rate of 10Hz.

3. TIME SERIES CLUSTERING

The algorithm we used to cluster time series data is a variation on the Segmental K-Means algorithm [6]. The procedure is as follows:

1. *Given:* N, the number of models, T the number of samples allocated to a state, S, the number of states per model, f the expected rate of class changes.

2. *Initialization:* Select N segments of the time series each of length T*S, spaced approximately $1/f$ apart. Initialize each of the N models with a segment, using linear state segmentation.

3. *Segmentation:* Compile the N current models into a fully-connected grammar. A nonzero transition connects the final state of every model to the initial state of every model. Using this network, resegment the cluster membership for each model.

4. *Training:* Estimate the new model parameters using the Forward-Backward algorithm on the segments from step 3. Iterate on the current segmentation until the models converge and then go back to step 3 to resegment. Repeat steps 3 and 4 until the segmentation converges.

We constrained ourselves to left-right HMMs with no jumps and single Gaussian states.

3.1. Time Hierarchy

Varying the frame-state allocation number directs the clustering algorithm to model the time-series at varying time scales. In the *Initialization* step, this time scale is made explicit by T, the frame-state allocation number, so that each model begins by literally modeling $S * T$ samples. Of course, the reestimation steps adaptively change the window size of samples modeled by each HMM. However, since EM is a local optimization the time scale will typically not change drastically from the initialization. Hence, by increasing the frame-state allocation we can build a hierarchy of HMMs where each level of the hierarchy has a coarser time scale than the one below it.

3.2. Representation Hierarchy

There are still important structures that just clustering at different time scales will not capture. For example, suppose we wanted a model for a supermarket visit, or a walk down a busy street. As it stands, clustering will only separate specific events like supermarket music, cash register beeps, walking through aisles, for the supermarket, and cars passing, crosswalks, and sidewalks for the busy street. It will not capture the fact that these events occur together to create scenes, such as the supermarket scene, or busy street scene. (Notice that simply increasing the time scale and model complexity to cover the typical supermarket visit is not feasible for the same reasons that speech is recognized at the phoneme and word level instead of at the sentence and paragraph level.)

We address this shortcoming by adapting a hierarchy of HMMs much a like a grammar. So beginning with a set of low-level hmms, which we will call object HMMs (like phonemes), we can encode their relationships into scene HMMs (like words). The process is as follows:

1. *Detect:* By using the Forward algorithm with a sliding window of length Δt, obtain the likelihood,
$$L_\lambda(t) = P(O_t, \cdots, O_{t+\Delta t}|\lambda)$$
for each object HMM, λ, at time, t.

2. *Abstract:* Construct a new feature space from these likelihoods,
$$F(t) = \begin{bmatrix} L_1(t) \\ \vdots \\ L_N(t) \end{bmatrix}$$

3. *Cluster:* Now cluster the new feature space into scene HMMs using the algorithm from Section 3.

test [1]

4. RESULTS

We evaluated our performance by noting the correlation between our emergent models and a human-generated transcription. Each cluster plays the role of a hypothesis. A hypothesis is verified when its indexing correlates highly with a ground truth labeling. Hypotheses that fail to correlate are ignored, but kept as "garbage classes". (Hence, it is necessary to have more clusters than "classes" in order to prevent the useful models from having to model everything.)

In the following experiments we restricted the system to two levels of representation (i.e. a single object HMM layer and a single scene HMM layer). The time scales were varied from 3 secs to 100 secs for the object HMMs, but kept at 100 secs for the scene layer.

Short Time Scale Object HMMs

In this case, we used a 3 sec time-scale for each object HMM and set the expected rate of class changes, f, to 30 secs. As a result, the HMMs modeled events such as doors, stairs, crosswalks, and so on. To show exactly how this worked, we give the specific example of the user arriving at his apartment building. This example is representative of the performance during other sequences of events. Figure 2 shows the features, segmentation, and key frames for the sequence of events in question. The image in the midde represents the raw feature vectors (top 81 are video, bottom are audio). Notice that you can even see the users steps in the audio spectrogram.

Long Time-scale Object HMMs

Here we increase the time-scale of the object HMMs to 100 secs. The results are that HMMs model larger scale changes such as long walks down hallways and streets.

We give some preliminary results for the performance of classification as compared to some hand-labeled ground truth. Since we did no training with labeled data, our models did not get the benefit of embedded training or garbage-modeling. Hence frequently the models are overpowered by a few that are not modeling anything useful. Typically this is where the system would make an application-driven decision to eliminate these models.

As an alternative we present the correlation coefficients between the the independently hand-labeled ground truth and the output likelihood of the highest correlating model. The table below shows the classes that the system was ably to reliably model from only 2hrs. of data:

Label	Correlation Coeff.
office	0.9124
lobby	0.7914
bedroom	0.8620
cashier	0.8325

Long Time-scale Scene HMMs

We also constructed a layer of scene HMMs that are based on the outputs of the Short Time-scale Object HMMs from above. Where before we were unable to clean classes for more complex events, like the supermarket visit and walk down a busy street, now this level HMMs is able to capture them. The following table gives the correlations for the best models:

Label	Correlation Coeff.
dorms	0.8024
charles river	0.6966
necco area	0.7495
sidewalk	0.7804
video store	0.9802

Figures 3 and 4 show the model likelihoods for the models that correlated with "walking down a sidewalk" and "at the video store". While the video store scene has elements that overlap with other scenes, the video store model is able to cleanly select only the visit to the video store.

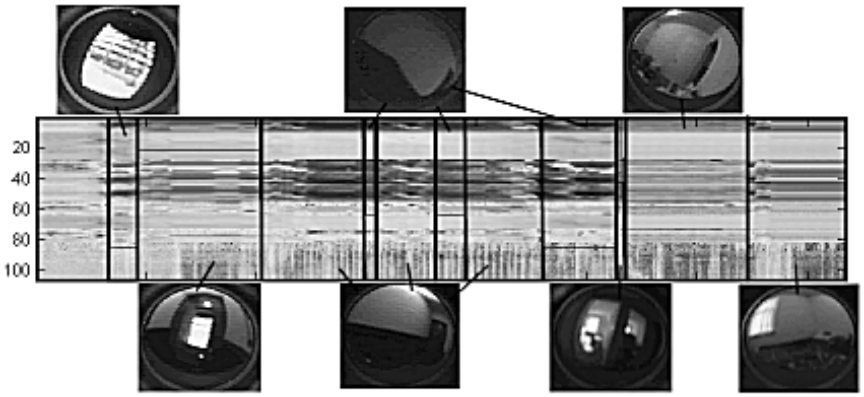

Figure 2: Coming Home: this example shows the user entering his apartment building, going up 3 stair cases and arriving in his bedroom. The system's segmentation is depicted by the vertical lines along with key frames.

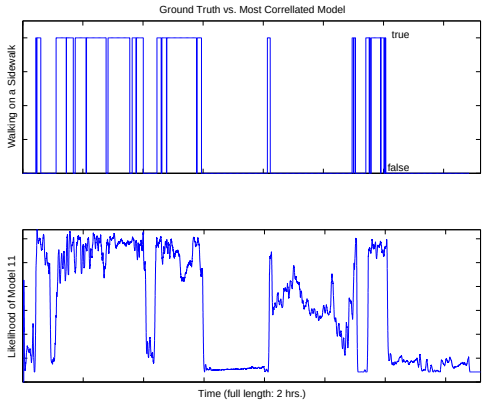

Figure 3: The Sidewalk Scene: above is the independently hand-labeled ground truth, below is the likelihood of the most correlated model.

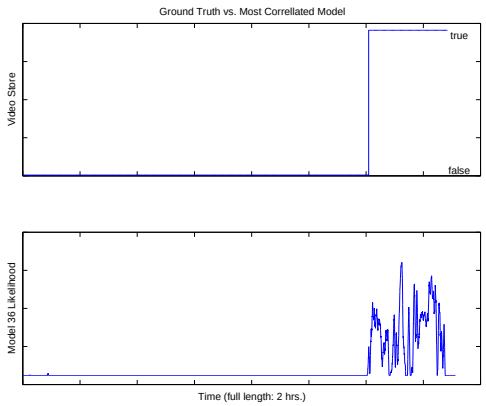

Figure 4: The Video Store Scene: above is the independently hand-labeled ground truth, below is the likelihood of the most correlated model.

5. CONCLUSION

It is pretty clear that the unsupervised clustering of audio/video data is feasible and useful. In addition, the clustering algorithm in this paper can be easily adapted to an incremental and pseudo-realtime framework. Instead of iterating over all past data, the system can have a "memory" by only training on a recent window of data. This implies that the system can then adapt as new memories habituate.

Our immediate goal is to integrate our system with a software agent so that the performance of our models can be grounded in some meaningful context.

6. REFERENCES

[1] Albert S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. The MIT Press, 1990.

[2] G. J. Brown. *Computational Auditory Scene Analysis: A representational approach*. PhD thesis, University of Sheffield, 1992.

[3] Bernhard Feiten and Stefan Gunzel. Automatic indexing of a sound database using self-organizing neural nets. *Computer Music Journal*, 1994.

[4] Liu, Wang, , and Chen. Audio feature extraction and analysis for multimedia content classification. *Journal of VLSI Signal Processing Systems*, 1998.

[5] Silvia Pfeiffer, Stephan Fischer, and Wolfgang Effelsberg. Automatic audio content analysis. Technical report, University of Mannheim, 1997.

[6] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 1989.

[7] Dan Siewiorek, editor. *The First International Symposium on Wearable Computers*, 1997.

[8] T. Starner, B. Schiele, and A. Pentland. Visual contextual awareness in wearable computing. In *Second International Symposium on Wearable Computers*, Oct 1998.