A SEMI-SUPERVISED LEARNING APPROACH TO ONLINE AUDIO BACKGROUND DETECTION

Selina Chu, Shrikanth Narayanan and C.-C. Jay Kuo

Department of Computer Science and Signal and Image Processing Institute
University of Southern California, Los Angeles, CA 90089-2564
{selinach, shri, cckuo}@sipi.usc.edu

ABSTRACT

We present a framework for audio background modeling of complex and unstructured audio environments. The determination of background audio is important for understanding and predicting the ambient context surrounding an agent, both human and machine. Our method extends the online adaptive Gaussian Mixture model technique to model variations in the background audio. We propose a method for learning the initial background model using a semi-supervised learning approach. This information is then integrated into the online background determination process, providing us with a more complete background model. We show that we can utilize both labeled and unlabeled data to improve audio classification performance. By incorporating prediction models in the determination process, we can improve the background detection performance even further. Experimental results on real data sets demonstrate the effectiveness of our proposed method.

Index Terms— Environmental sounds, unstructured audio classification, background modeling, semi-supervised learning.

1. INTRODUCTION

The ability to automatically characterize general audio environment types is an important step toward developing the understanding of a scene (or context) surrounding an audio sensor [1–4]. To arrive at a better understanding of the audio context or scene, we investigate the modeling of background audio of complex environments. Unstructured acoustic context refers to a location or setting with a variety of different acoustic characteristics such as a coffee shop, park or street. Unlike speech and music, which have formantic structures and harmonic structures, respectively, unstructured audio is variably composed from different heterogeneous sound sources, and hence can have diverse signal characteristics. There are several motivating applications that can benefit from context awareness. Examples include mobile device-based services and wearable interfaces, as well as providing hearing for robots or surveillance systems, which are currently predominantly vision based. The first step toward this direction is the ability to comprehend the unstructured ambient context. Being able to enhance the system's context awareness by incorporating audio information, along with existing visual knowledge, would have significant utility.

The background in an ambient auditory scene can be considered as something recurring, and noise-like, which is made up of various sound sources, but changing over time, *i.e.*, traffic and passers-by on a street. In contrast, the foreground can be viewed as something unanticipated or as a deviation from the background model, *i.e.*, passing ambulance with siren. The problem arises when identifying foreground existence in the presence of background noise, given the background also changes with a varying rate, depending on different environments. If we create fixed models with too much prior knowledge, these models could be too specific and might not do well with new sounds. On the other hand, models that do not consider prior knowledge, such as unsupervised techniques, typically use simple methods of thresholding [5]. Then, we would be led to the problem of threshold determination. This would be impractical in unstructured environments since there are no clear boundaries between different types of sounds.

Systems employing learning techniques, such as in [6], build models explicitly for specific audio events, making it inflexible to new events. The state-of-the-art approaches in background modeling [7] do not make any assumptions about the prior knowledge of the location and operate with ephemeral memory of the data. The method proposed in [7] models the persistency of features by defining the background as changing slowly over time and assuming foreground events to be short and terse, *e.g.*, breaking glass. The problem arises when the foreground is also gradual and longer lasting, *e.g.*, plane passing overhead. In this case, it would adapt the foreground sound as background, since there is no knowledge of the background or foreground. It would be difficult to verify whether the background model is indeed correct or models some persistent foreground sound as well.

In this work, we consider modeling and detecting background and foreground sounds by incorporating explicit knowledge of data into the process. We propose to include audio prediction models as a procedure to learn the background and foreground sounds. Our framework is comprised of two modules, each addressing a separate issue. First, we use a semi-supervised method to train classifiers to learn models for the foreground and background of an environment. Then, we use the learned models as a way to bootstrap the overall system. A separate model is constructed to detect the changes in the background. It is then integrated together with audio prediction models to decide on the final background/foreground (BG/FG) determination.

The rest of this paper is organized as follows. Sec. 2 examines the semi-supervised learning approach applied to audio data. Sec. 3 describes a framework for background modeling. Our experimental data and setup are presented in Sec. 4 and experimental results and discussion are given in Sec. 5. Finally, concluding remarks are provided in Sec. 6.

2. SEMI-SUPERVISED LEARNING WITH AUDIO

We begin our study by building prediction models to classify the enivironment into foreground and background. To obtain classifiers with high generalization ability, a large amount of training samples

Table 1. Classification results using self-training. (in %)

Data set	EM	EM-λ
Coffee room	88.7	92.5
Courtyard	76.1	81.0
Subway platform	73.7	86.5

are typically required. However, labeled samples are fairly expensive to obtain while unlabeled natural recordings consisting of environmental sounds are easy to come by. Thus, we investigate ways to automatically label them using self-training [8] to increase the training example size. In self-training, a classifier for each class is trained on a labeled data set. Then, it *labels* these unlabeled examples automatically. The newly labeled data are added to the original labeled training set, and the classifier is refined with the *augmented* data set.

After the above steps, we train a multivariate probability density model to recognize the background and a separate one for the foreground, where the expectation maximization (EM) approach is used to estimate the parameters of the model. We use the augmented EM approach (EM-λ) in [8], where parameters are estimated using both labeled and unlabeled samples. The contribution of unlabeled samples are weighted by a factor $0 \leq \lambda \leq 1$, which provides a way to reduce the influence of the use of a large amount of unlabeled data (as compared to the labeled data). It also makes the algorithm less sensitive to the newly *labeled* data. With the standard EM, we maximize the M-step using:

$$l_c(\theta|X) = log(P(\theta)) + \sum_{x_i \epsilon X} log \sum_{i=1}^{K} \alpha_j P(X_i|\theta_j), \quad (1)$$

where there are N training data, X, with class labels $y_i \epsilon K$.

When unlabeled data $x_i \epsilon X^u$ are incorporated into the labeled data $x_i \epsilon X^l$, the new training set becomes $X = X^l \cup X^u$. Then, to maximize the M-step in EM-λ, we have

$$\begin{aligned} l_c(\theta|X) = log(P(\theta)) + \sum_{x_i \epsilon X^l} log \sum_{i=1}^{K} \alpha_j P(X_i|\theta_j) \\ + \lambda \left(\sum_{x_i \epsilon X^u} log \sum_{i=1}^{K} \alpha_j P(X_i|\theta_j) \right), \end{aligned} \quad (2)$$

which results in the following parameter estimation:

$$P(X|k_j;\theta) = \frac{1 + \sum_{i=1}^{|X|} P(y_i = k_j|x_i)}{\sum_{i=1}^{|X|} \Lambda(i) P(y_i = k_j|d_i)} \quad (3)$$

with the prior as

$$P(k_j|\theta) = \frac{1 + \sum_{i=1}^{|X|} \Lambda(i) P(y_i = k_j|x_i)}{|K| + |X^u| + \lambda |X^u|}, \quad (4)$$

and a weighting factor $\Lambda(i)$, defined as

$$\Lambda(i) = \begin{cases} \lambda, & \text{if } x_i \epsilon D^u, \\ 1, & \text{if } x_i \epsilon D^l. \end{cases} \quad (5)$$

If none of the classifiers found the unlabeled sample to be probable (*e.g.*, probabilities are low, say, less than 15%), we assign the unlabeled data to the foreground classifier since it is more likely that the unseen data sample is part of the foreground model.

To demonstrate the effectiveness of this semi-supervised training approach for our dataset, we compare results between the usual EM approach and the EM-λ approach. The experimental setup is described in Sec. 4. After using the self-training method to label a large amount of unlabeled audio data, we re-train models P_{fg} and P_{bg} for the background determination process. Experimental results are summarized in Table 1.

3. ONLINE ADAPTIVE BACKGROUND DETECTION

Once prediction models, P_{fg} and P_{bg}, are learned for foreground (FG) and background (BG) classification, we utilize these models in the online adaptive background detection process. The initial background modeling work was done for video [9], which uses the mixture of Gaussians for each pixel. Instead of modeling the *pixel process*, Moncrieff *et al.* [7] propose to model the audio feature vector as Gaussians mixture densities. Our adaptation is based on the latter. The resultant algorithm is summarized below.

The history of feature vector x_t can be viewed as $\{x_1, x_2, \ldots, x_t\}$, each x_t is modeled by a mixture of K Gaussian distributions. The probability of observing current x_t is given by

$$P_{online}(x_t) = \sum_{i=1}^{K} \alpha_{i,t} P(x_t|\theta_{i,t}).$$

That is, x_t is represented by the components of the mixture model. Since x_t varies over time, the mixture models have to be re-trained at every time t to maximize the likelihood of X. Instead, we use an online K-means approximation algorithm. Every x_t is checked against the existing K Gaussian distributions to determine if a match is found. The K^{th} component is viewed as a match if x_t is within 2.5 standard deviations from the mean of a distribution, as done in [7,9]. If none of the distributions qualify, the least probable distribution is replaced by the current observation x_t as the mean value with an initial high variance and a low prior. The parameters are adjusted with the prior weights of each component as

$$\alpha_{k,t} = (1 - \beta_\omega * M_{k,t})\omega_{k,t-1} + \beta_\omega(M_{k,t}), \quad (6)$$

where $M_{k,t}$ is 1 for matched models and 0 for mismatched ones and β_ω is the learning rate, which determines the rate of adaptation of the background model. After the approximation, priors are re-normalized (summing to 1) and used to decrease the weight of the models that are not matched. The parameters for unmatched models remain the same, while matched models update their parameters with new observation x_t as

$$\mu_{k,t} = (1 - \rho)\mu_{t-1} + \rho X_t$$

$$\Sigma_t^{i,j} = (1 - \rho)\Sigma_{t-1}^{i,j} + \rho(X_t^i X_t^j),$$

where

$$\rho = \beta_g e^{-\frac{1}{2d}(X_t - \mu_{t-1})^T \sum_{t-1}^{-1} (X_t - \mu_{t-1})}$$

is the second learning rate that is dependent on the data and β_g determines the update rate of model parameters. Using this method to update does not destroy the existing models with new incoming data, but remains in the overall system (while having their prior weights α decrease). The model with the lowest α becomes the the least probable model, which will then be replaced by the new observation.

From here on, we deviate from that of [7] by including both P_{fg} and P_{bg} into the system. For updating P_{fg} and P_{bg}, we continue to use Eqs. (2)-(4), but in a sliding windows approach. As the system runs online, if we permit an increasing amount of unlabeled data to be included in the model, it would allow errors from the self-training process to propagate. To regulate possible concept drifts (as trends and patterns tend to change over time), we use a simple sliding window method, where only unlabeled data within the window of size m is utilized in the self-training process. Any unlabeled data outside the window is not used. Furthermore, to avoid re-training models at every second, we perform the self-training process at every $\frac{m}{4}$ interval. This means that we remove the oldest $\frac{m}{4}$ of the data from the window to include newly arrived unlabeled data. For example if m = 120 samples, then we perform self-training at every 30 samples interval.

In addition, we attempt to maintain P_{fg} to reflect the current concept by placing more emphasis on the current data sample and using a new Λ_{fg} for P_{fg} defined as

$$\Lambda_{fg}(i) = \begin{cases} \frac{m-z_m}{m}, & \text{if } x_i \epsilon D^u, \\ \lambda, & \text{if } x_i \epsilon D^l, \end{cases} \quad (7)$$

where $z_i = \{z_1, z_2, \ldots, z_m\}$, and where z_1 and z_m are the beginning and the end of window m, respectively.

For BG/FG classification, we rank their distributions by prior weights $\bar{\alpha}_{online}$ and α_{bg}, α_{fg} (from P_{bg}, P_{fg}, respectively, depending on $P(k|x_t;\theta)$). Since we use separate models for the foreground and background, we normalize their priors to 1. For classification, we ordered the Gaussians by their values of $\alpha_{online} + \alpha_s$, where $\alpha_s = \alpha_{bg}$ if $P_{bg}(\mu_t) \geq P_{fg}(\mu_t)$ and 0 otherwise. The distributions chosen as the foreground model are obtained via

$$FG = \left[\sum_{k=1}^{k_{highest}} \alpha_{online_k} + \alpha_{s_k} \right] \leq T,$$

where $k_{highest}$ is the highest rank model and $k = 1$ the lowest ranked. T is the tolerance threshold for background classification. A lower T will result in more distributions being classified as background. Models not chosen as foreground are consider background models.

We use a heuristic to perform the final classification. We utilize a queue to keep track of results at each time t from either a background or foreground classification of $P_{bg|fg}(x_t, x_{t-1}, \ldots)$. If there is no change between classifications of previous distribution P^{t-1}_{online} and the current one, P^{t}_{online}, we append the result $P_{bg|fg}(x_t)$ into the queue and make the classification at t by taking a majority vote of the result queue at $t, t-1, \ldots, t-q$, where q is the queue size. Using a queue to *remember* the results allows for some misclassification in the prediction process. When there is change between P^{t-1}_{online} and P^{t}_{online}, we examine $P_{bg|fg}(x_t)$ and $P_{bg|fg}(x_{t-1})$. If they are consistent, we also take a majority vote. Otherwise, we take on the classification $P_{bg|fg}(x_t)$ at t. Whenever there is a change in classification, we clear the queue of results from $t-1, \ldots, t-q$.

4. DATA AND EXPERIMENTS

To demonstrate the effectiveness of the proposed online background modeling algorithm, the following three environment sounds are used in the test:

1. Coffee room: Background includes footsteps, shuffling of things, people coming in and out. Foreground includes coffee grinding and brewing, printing sound (since a printer is located in the coffee room), etc.;
2. Courtyard: Background includes water fountain, distant talking from passers-by and traffic from nearby streets. Foreground includes plane passing overhead, cellphone ringing, loud talking, footsteps, etc.;
3. Subway station platform: Background includes passers-by noise and talking, trains in the distant. Foreground includes train arrival/departure, trains breaking, announcements, etc.

They are made up of ambient noise of a particular environment, composed of many sound events. We do not consider each constituent sound event individually, but as many properties of each environment. We use continuous, unedited audio streams as the training and testing data. The first two data sets were collected and recorded in mono-channel, 16 bits per sample with a sampling rate of 44.1 KHz and of varying lengths in the Electrical Engineering building of the University of Southern California. They were taken at various times over a period of two weeks, with a duration averaging around 15 minutes each. The *subway station* set is also recordings of natural (unsynthesized) sound clips, which were obtained from [10] and down-sampled to a 22.050 KHz sampling rate.

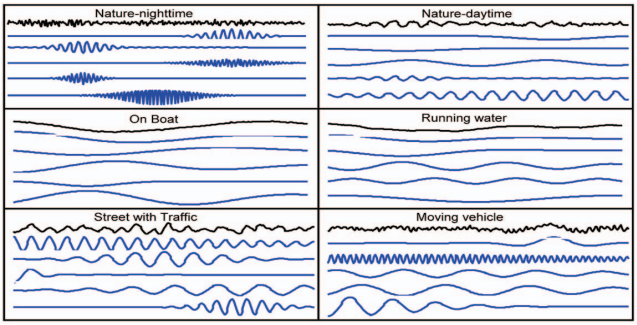

Fig. 1. Decomposition of signals from 6 environments, where the top-most signal is the original, followed by the first five basis vectors, demonstrating different underlying structures for various environments, where an MP-based algorithm picks up these top basis vectors and represents them uniquely.

The incoming audio signal was segmented into fixed duration 1-second clips. Every second of the data sets was manually labeled. Features were computed for every 1-second window by averaging those from a 30-msec rectangular sampling window with 5 msec overlap. They were calculated for each clip and combined to form those of current clip x_t. We use two types of features: 1) Mel-frequency cepstrum coefficient analysis (MFCC), widely popular in speech and audio processing, and 2) MP-features [4]. MP-features utilize the matching pursuit (MP) algorithm and a dictionary of Gabor atoms to learn the inherent structure of each type of sounds and select a small set of joint time frequency features. This method has shown to be robust for classifying sounds where the pure frequency-domain features fail and can be advantageous in combining with MFCC to improve the overall performance. Examples of MP-features are given in Fig. 1. The feature vector used in this work contains a combination of MFCC and MP-features. For details of feature extraction and extensive experiments, we refer to [11].

For evaluation, the BG/FG classification was compared with labeled testing sets. We had 40 minutes of data for *Coffee room* and *Courtyard*. We were only able to obtain 15 minutes for the *Subway station*. The data were divided into 4 sets for each class. We used 2 sets as unlabeled data and 1 set as labeled in the self-training process. The last subset was used for testing in the final online BG/FG determination. Results were taken from the average of six trials (from different permutations of the 4 sets). The data were segmented into 1-second segments, but analyzed in sequence, where each segment was considered a sample. The accuracy of the detection is calculated by

$$\textit{BG accuracy} = \frac{N_{y=bg}}{N_{total} - N_{fg}},$$

where $N_{y=bg}$ is the number of samples classified as BG, N_{fg} is the number of FG samples that are correctly classified, and N is the total number of samples.

We calibrated the parameter values for each dataset to produce better overall results. The weighting factor, $\lambda = 0.5$, was set to reduce the sensitivity to unlabeled data. The threshold, $T = 0.5$, was the tolerance for determining distributions that were considered as BG. α_g and α_ω were set to 0.01 in the experiments. The sliding window size m was set to 120 samples. Based on the observation

from [4], the setting for the MP-features was chosen for the Gabor function with the following parameters in this work: $s = 2^p$ ($1 \leq p \leq 8$), $u = \{0, 64, 128, 192\}$, $\omega = Ki^{2.6}$ (with $1 \leq i \leq 35$, $K = 0.5 \times 35^{-2.6}$ so that the range of ω is normalized between 0 and 0.5), $\theta = 0$ and the atom length is truncated to $N = 256$. Thus, the dictionary consists of $1120 = 8 \times 35 \times 4$ Gabor atoms that were generated using scales of 2^p and translated by quarters of atom length N.

5. RESULTS AND DISCUSSION

We compared the performance accuracy between the proposed method and the one from [7] as a baseline. We refer to our approach as *combination models* (CM) and [7] as *persistency only models* (PSM). The experimental results are summarized in Table 2. We see that both methods produce better accuracy for the background than foreground since the background is more constant than the foreground, and therefore is easier to learn. The PSM method performs poorly on *Coffee room* data since it cannot classify the long-persistent sound of a printer as the foreground. At one time, the printing sound continuously ran for 49 seconds and these segments were misclassified as background.

Table 2. Background detection accuracy (in %)

	CM		PSM	
Data	FG	BG	FG	BG
Coffee room	75.9	82.5	27.4	56.8
Courtyard	63.5	92.1	36.7	89.9
Subway platform	74.8	79.2	46.5	58.9

We examine a small segment of data (as shown in Fig. 2) in more detail. In this example, the delay from PSM was about 7 seconds, while CM results in a 2-3 second delay. We also note that, after about 10 seconds of considering the current sound clip as foreground, the foreground distributions were soon considered to part of the background process. With a quick change in the BG/FG events, PSM takes about 10-15 seconds to stabilize depending on the update rates.

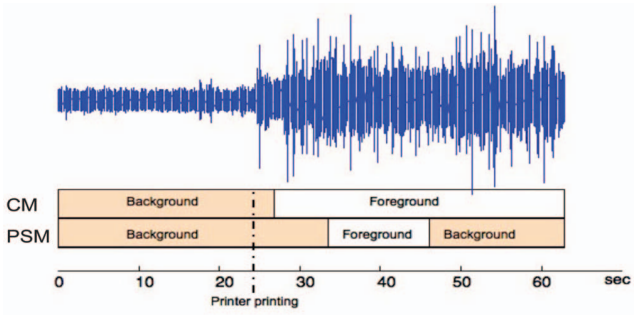

Fig. 2. Comparison of classification results obtained by online background modeling with *combination models* (CM) and *persistency only models* (PSM).

We observe that it is more difficult to detect the foreground segments in the *Courtyard* class. When a plane passed over for 16 seconds, PSM only detected 4 seconds of it while CM detected about 10 seconds. The *Subway* set provides an example comprised of many short events. There were very few moments when there is a constant background. In this case, we observe that it was difficult for both systems to achieve high performances. However, CM still outperforms PSM. For the CM method, class determination is based on the combined effort of both online models and prediction models, making it less sensitive to changes in parameters. The PSM method is more sensitive to parameter changes since its classification only depends on one model.

6. CONCLUSION AND FUTURE WORK

In this work, we proposed a framework for audio background modeling, which includes prediction, data knowledge and persistent characteristics of the environment, leading to a more robust audio background detection algorithm. This framework has the ability to model the background and detect foreground events as well as the ability to verify whether the predicted background is indeed the background or a foreground event that protracts for a longer period of time. Experimental results demonstrated promising performance in improving the state-of-the-art in background modeling of audio environments. We also investigated the use of a semi-supervised learning technique to exploit unlabeled audio data. It is encouraging that we could utilize more unlabeled data to improve generalization as they are usually cheap to acquire but expensive to label. And more than often, we are forced to work with relatively small amount of labeled data due to this limitation. Future work will include using an ensemble of classifiers to tackle the problem of concept drifts, *e.g.,* having a different classifier for each recent time period and adaptively learning the window size to the current extent of the concept drift.

7. REFERENCES

[1] J. Huang, "Spatial auditory processing for a hearing robot," in *Proc. of ICME*, 2002.

[2] D. P. W. Ellis and K. Lee, "Minimal-impact audio-based personal archives," in *Proc. of CARPE*, 2004.

[3] A. Eronen, V. Peltonen, J. Tuomi, A. Klapuri, S. Fagerlund, Timo Sorsa, Gaetan Lorho, and Jyri Huopaniemi, "Audio-based context recognition," *IEEE Trans. on Audio, Speech and Lang. Processing*, 2006.

[4] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition using MP-based features," in *Proc. of ICASSP*, 2008.

[5] A. Härmä, M. McKinney, and J. Skowronek, "Automatic surveillance of the acoustic activity in our living enivronment," in *Proc. of ICME*, 2005.

[6] D. P. W. Ellis, "Detecting alarm sounds," in *Proc of workshop on CRAC*, 2001.

[7] S. Moncrieff, S. Venkatesh, and G. West, "Online audio background determination for complex audio environments," *ACM Trans on Multimedia Computing, Communications and Applications*, vol. 3, no. 2, pp. 1–30, 2007.

[8] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine Learning*, vol. 39, pp. 103–134, 2000.

[9] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings of CVPR*, 1999.

[10] "The freesound project," http://freesound.iua.upf.edu/index.php.

[11] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Trans. on Audio, Speech and Lang. Processing*, In press, 2009.