EARLY EVENT DETECTION IN AUDIO STREAMS

Huy Phan[⋆†], *Marco Maass*[⋆], *Radoslaw Mazur*[⋆], and *Alfred Mertins*[⋆]

[⋆]Institute for Signal Processing, University of Lübeck, Germany
[†]Graduate School for Computing in Medicine and Life Sciences, University of Lübeck, Germany
{phan, maass, mazur, mertins}@isip.uni-luebeck.de

ABSTRACT

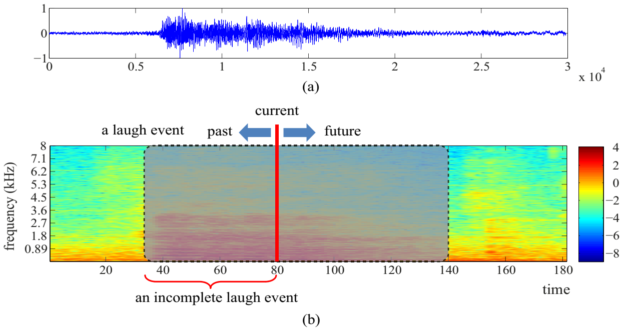
Audio event detection has been an active field of research in recent years. However, most of the proposed methods, if not all, analyze and detect complete events and little attention has been paid for early detection. In this paper, we present a system which enables early audio event detection in continuous audio recordings in which an event can be reliably recognized when only a partial duration is observed. Our evaluation on the ITC-Irst database, one of the standard database of the CLEAR 2006 evaluation, shows that: on one hand, the proposed system outperforms the best baseline system by 16% and 8% in terms of detection error rate and detection accuracy respectively; on the other hand, even partial events are enough to achieve the performance that is obtainable when the whole events are observed.

***Index Terms*—** Early detection, audio event detection, online detection, regression forests

1. INTRODUCTION

Recognizing audio events from audio streams arises in a variety of applications ranging from ambient intelligence [1, 2] to surveillance [3, 4]. A temporal event has a duration, and by early detection, as in [5], we mean to detect the event as soon as possible, after it starts but before it ends. This idea is illustrated in Fig. 1. In many situations, the reliable early detection of the target events is crucial, because without it, the intended application would fail. As in the example from [5], when one wants to build a robot that interacts with humans, reliable and rapid event detection is a key requirement so that the robot can make appropriate responses in a timely manner. Otherwise, the responses would be out of synchronization. For another application in which a camera surveillance system is guided by audio event detection [4], the system needs to detect the events and take actions as fast as possible. Directing the cameras after the events are already completed may be too late, as the objects already moved. In general, the earliness of detection without scarification of the accuracy is always preferable.

Despite the importance of early detection, little attention has been paid. Recently, a few works were explicitly proposed in other fields (e.g. computer vision [5, 6]). Past research in audio event detection can be roughly classified into two approaches: detection-by-classification [2, 7] and HMM-based joint classification/segmentation [8, 9]. Nevertheless, they were interested in detecting complete events and no reports on the perspective of early detection has been found. Furthermore, from analysis in [5], the detectors based on these methods are usually trained to recognize complete events only and they require observing the entire event for a reliable decision. Consequently, using them for online early detection, which requires the ability to recognize a partial event, would result in unreliable decisions. The reason is that reliable early detection requires a monotonically growing detection scoring function [5] that is not easy to obtain with a simple solution based on these methods.



(a)

(b)

Fig. 1. Can we even detect a laugh event before it finishes? (a) waveform and (b) spectrogram.

Although the structured output SVM framework [5] can be adapted for early audio event detection, we alternatively propose a learning formulation with decision forests framework [10]. Motivated by the works of Phan *et al.* [11, 12], in which the joint event detection/segmentation is posed as a regression problem and resolved with random regression forests, we re-formulate the joint problem to accommodate sequential data for early detection. We simulate an audio stream as sequential superframe-by-superframe data arrival and detect the events correctly when their partial durations are observed. We further prove the monotonicity of the detection scoring function derived from our formulation. The proposed system also obtains the temporal extents of the detected events and provides an efficient event tracking mechanism as joint results. Compared to the structured output SVM framework [5], our formulation based on decision forests offers numerous advantages. First, it is unnecessary to augment the training process with partial events which exponentially grows the size of the training data. Second, our system does not perform quadratic temporal scale search for detection. Last but not least, the monotonicity of the scoring function in [5] may be no longer valid for periodic events, which is common for audio events, but it is not the case in our formulation.

The experimental results on the ITC-Irst database shows that while the proposed system inherits the state-of-the-art performance, it always makes faster detection on all target event categories without losing the detection accuracy. In comparison with the common com-

This work was supported by the Graduate School for Computing in Medicine and Life Sciences funded by Germany's Excellence Initiative [DFG GSC 235/1]

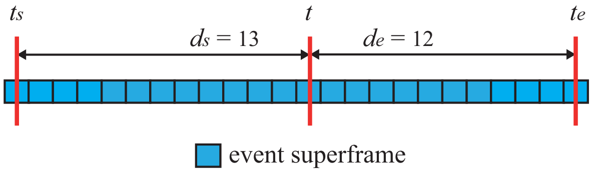


Fig. 2. The onset displacement d_s and the offset displacement d_e of a superframe at the time t to the event onset t_s and offset t_e.

peting methods (which can only run in offline mode), the proposed system significantly outperforms all of them in terms of detection error rate and detection accuracy.

2. RANDOM REGRESSION FORESTS FOR EVENT ONSET AND OFFSET ESTIMATION

2.1. Training

The construction of the regression forests for a target event category is supervised, following the common decision forests framework [10]. The isolated events in training data are divided into interleaved *superframes* [11] (the description of a superframe will be entailed in Section 4.1) to obtain the set of annotated superframes $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{d}_i)\}$. A superframe is represented by $\mathbf{x} \in \mathbb{R}^M$ where M is the dimensionality. $\mathbf{d} = (d_s, d_e) \in \mathbb{R}_+^2$ denotes the displacement vector (in superframes) of the superframe to the event onset t_s and offset t_e inclusive, as illustrated in Fig. 2. The onset displacement d_s and offset displacement d_e are computed as

$$d_s = t - t_s, \tag{1}$$

$$d_e = t_e - t. \tag{2}$$

In order to construct a tree of the regression forest $\mathcal{F}$, a subset of superframes is randomly sampled from $\mathcal{S}$. Starting from the root, at a split node ℓ a set of binary tests $t_{f,\tau}$ defined in (3) is generated.

$$t_{f,\tau}(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x}^f > \tau \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Here, $\mathbf{x}^f$ denotes the value of $\mathbf{x}$ at the random selected feature channel $f \in \{1, \ldots, M\}$. The variable τ is a random threshold generated in the range of $\mathbf{x}^f$. An optimal test is then adopted from the test set to split the superframe set $\mathcal{S}_\ell$ at the split node ℓ into two sets $\mathcal{S}_\ell^{\text{right}}$ and $\mathcal{S}_\ell^{\text{left}}$:

$$\mathcal{S}_\ell^{\text{right}} = \{(\mathbf{x}, \mathbf{d}) \in \mathcal{S}_\ell \big| t_{f,\tau}(\mathbf{x}) = 1\}, \tag{4}$$

$$\mathcal{S}_\ell^{\text{left}} = \{(\mathbf{x}, \mathbf{d}) \in \mathcal{S}_\ell \big| t_{f,\tau}(\mathbf{x}) = 0\}. \tag{5}$$

$\mathcal{S}_\ell^{\text{right}}$ and $\mathcal{S}_\ell^{\text{left}}$ are subsequently sent to the right and the left child nodes, respectively. The adoption criteria is to minimize the *displacement uncertainty* U:

$$U = \sum \big\|\mathbf{d}_i^{\text{left}} - \bar{\mathbf{d}}^{\text{ left}}\big\|_2^2 + \sum \big\|\mathbf{d}_i^{\text{right}} - \bar{\mathbf{d}}^{\text{ right}}\big\|_2^2. \tag{6}$$

Here, $\bar{\mathbf{d}}$ denotes the mean displacement vector of the corresponding superframe set indicated by the superscript. By this, the superframes are clustered by both their features and their relative positions to event onsets and offsets.

The splitting process is recursively repeated until the maximum depth $D_{\max}$ is reached or a minimum number of superframes $N_{\min}$ is remained. Then a leaf node will be created. The displacement vectors of the remaining superframes at the leaf node are modeled and stored as a two-dimensional Gaussian distribution $\mathcal{N}(\mathbf{d}\big|\bar{\mathbf{d}}, \mathbf{\Gamma})$:

$$\mathcal{N}(\mathbf{d}\big|\bar{\mathbf{d}}, \mathbf{\Gamma}) = \frac{1}{2\pi\sqrt{\det(\mathbf{\Gamma})}} \exp\big(-\frac{1}{2}(\mathbf{d}-\bar{\mathbf{d}})^T \mathbf{\Gamma}^{-1}(\mathbf{d}-\bar{\mathbf{d}})\big). \tag{7}$$

where $\bar{\mathbf{d}} = (\bar{d}_s, \bar{d}_e)$ and $\mathbf{\Gamma} = \begin{pmatrix} \Gamma_s & 0 \\ 0 & \Gamma_e \end{pmatrix}$ are, respectively, the mean and the covariance matrix of the displacement vectors. However, for simplicity we do not consider covariance between onset and offset displacements. That is, $\mathcal{N}(\mathbf{d}|\bar{\mathbf{d}}, \mathbf{\Gamma})$ is equivalent to two univariate Gaussian distributions $\mathcal{N}_s(d|\bar{d}_s, \Gamma_s)$ and $\mathcal{N}_e(d|\bar{d}_e, \Gamma_e)$:

$$\mathcal{N}_s(d\big|\bar{d}_s, \Gamma_s) = \frac{1}{\sqrt{2\pi\Gamma_s}} \exp\big(-\frac{(d-\bar{d}_s)^2}{2\Gamma_s}\big), \tag{8}$$

$$\mathcal{N}_e(d\big|\bar{d}_e, \Gamma_e) = \frac{1}{\sqrt{2\pi\Gamma_e}} \exp\big(-\frac{(d-\bar{d}_e)^2}{2\Gamma_e}\big). \tag{9}$$

The above algorithm is repeated to grow all the trees in the forest $\mathcal{F}$.

2.2. Testing

Given a test superframe $\mathbf{x}$, we aim at estimating its displacements from the onset and offset of a target event using the learned regression forest $\mathcal{F}$. We input $\mathbf{x}$ into a tree $\mathcal{T}_i$ of $\mathcal{F}$. At each split node, the stored binary test is evaluated on $\mathbf{x}$, directing it either to the right or left child until ending up at a leaf node ℓ_i. From (8) and (9), estimates of the onset and offset displacements are obtained in terms of the Gaussian distributions stored at ℓ_i:

$$p_{d_s}(d|\ell_i, \mathbf{x}) = \mathcal{N}_s(d\big|\bar{d}_s^{\ell_i}, \Gamma_s^{\ell_i}), \tag{10}$$

$$p_{d_e}(d|\ell_i, \mathbf{x}) = \mathcal{N}_e(d\big|\bar{d}_e^{\ell_i}, \Gamma_e^{\ell_i}). \tag{11}$$

The posterior probabilites are finally computed by summing up $p_{d_s}(d|\ell_i, \mathbf{x})$ and $p_{d_e}(d|\ell_i, \mathbf{x})$ over all trees of the forest $\mathcal{F}$:

$$p_{d_s}(d|\mathbf{x}) = \frac{1}{|\mathcal{F}|} \sum_i \mathcal{N}_s(d\big|\bar{d}_s^{\ell_i}, \Gamma_s^{\ell_i}), \tag{12}$$

$$p_{d_e}(d|\mathbf{x}) = \frac{1}{|\mathcal{F}|} \sum_i \mathcal{N}_e(d\big|\bar{d}_e^{\ell_i}, \Gamma_e^{\ell_i}). \tag{13}$$

Here $\big|\mathcal{F}\big|$ denotes the number of trees of the forest $\mathcal{F}$. The expectations of $p_{d_s}(d|\mathbf{x})$ and $p_{d_e}(d|\mathbf{x})$, respectively, indicate the onset and offset displacements estimated by the superframe $\mathbf{x}$.

2.3. Inference

In online scenarios, we want to estimate where in time an event starts and ends in a continuous audio signal. Let t and t' both denote the time index. From (12) and (13), an event superframe $\mathbf{x}_{t'}$ at the time t' gives estimates of the onset and offset displacements as

$$p_{d_s}(d|\mathbf{x}_{t'}) = \frac{1}{|\mathcal{F}|} \sum_i \mathcal{N}_s(d\big|\bar{d}_s^{\ell_i}, \Gamma_s^{\ell_i}), \tag{14}$$

$$p_{d_e}(d|\mathbf{x}_{t'}) = \frac{1}{|\mathcal{F}|} \sum_i \mathcal{N}_e(d\big|\bar{d}_e^{\ell_i}, \Gamma_e^{\ell_i}). \tag{15}$$

From (1) and (2), estimates for the onset and offset positions are then obtained by placing $\mathcal{N}_{d_s}(d\big|\bar{d}_s^{\ell_i}, \Gamma_s^{\ell_i})$ in (14) at $\bar{d}_s^{\ell_i}$ backward

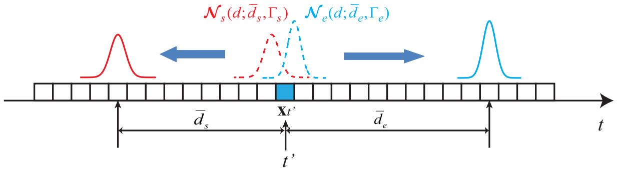

$\mathcal{N}_s(d;\bar{d}_s,\Gamma_s)$
$\mathcal{N}_e(d;\bar{d}_e,\Gamma_e)$
$\bar{d}_s$
$\mathbf{x}_{t'}$
$\bar{d}_e$
t
t'

Fig. 3. Estimation for event onset and offset positions: placing the onset displacement Gaussian backward and the offset displacement Gaussian forward from t'.

from t' and $\mathcal{N}_{d_e}(d|\bar{d}_e^{\ell_i}, \Gamma_e^{\ell_i})$ in (15) at $\bar{d}_e^{\ell_i}$ forward from t':

$$p_{t_s}(t|\mathbf{x}_{t'}) = \frac{1}{|\mathcal{F}|}\sum_i \mathcal{N}_s(t|t' - \bar{d}_s^{\ell_i}, \Gamma_s^{\ell_i}), \tag{16}$$

$$p_{t_e}(t|\mathbf{x}_{t'}) = \frac{1}{|\mathcal{F}|}\sum_i \mathcal{N}_e(t|t' + \bar{d}_e^{\ell_i}, \Gamma_e^{\ell_i}). \tag{17}$$

This step is illustrated in Fig. 3. The estimates by all superframes are accumulated to yield the confidence scores that the onset and offset positions of the target event coincide at a time t:

$$f_s(t) = \sum_{t'} p_{t_s}(t|\mathbf{x}_{t'}), \tag{18}$$

$$f_e(t) = \sum_{t'} p_{t_e}(t|\mathbf{x}_{t'}). \tag{19}$$

Ideally, if there exists only one event instance in the signal, its onset and offset positions can be determined as:

$$\hat{t}_s = \arg\max_t f_s(t), \tag{20}$$

$$\hat{t}_e = \arg\max_t f_e(t). \tag{21}$$

However, an audio stream typically contains multiple event occurrences, one after another, which must be detected sequentially. We propose an online detection mechanism in the Section 3.3.

3. EARLY AUDIO EVENT DETECTION SYSTEM

3.1. Early audio event detection on audio streams

For early event detection from the audio streams which come in sequentially, superframe-by-superframe, we mean to detect the event as soon as possible before it finished. Fortunately, as can be seen from (18) and (19), our system accommodates for sequential data very well. On another hand, our scoring functions show the monotonicity property, i.e. the confidence scores will increase as long as we observe more superframes. While this property is essential for reliable early detection [5], it cannot be assured by a naive solution that simply detects a partial event.

Without loss of generality, let us assume a sequence of superframes of length L starting from t_0. The sequence contains only one target event starting at t_1 and ending at t_2, where $t_0 < t_1, t_2 < L$. Furthermore, let $\max_{t_{cur}}(f_s)$ denote the maximum onset confidence score accumulated up to the time t_{cur}, where $t_0 \leq t_{cur} < L$ and f_s is given in (18). We have

$$\max_{t_{cur}}(f_s) = \max\Big(\sum_{t'=t_0}^{t_{cur}} p_{t_s}(t|\mathbf{x}_{t'})\Big). \tag{22}$$

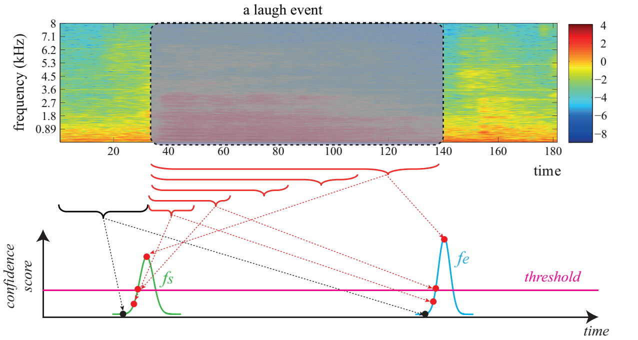


Fig. 4. When both accumulating f_s and f_e reach the threshold, the target event is considered detected.

The position corresponding to $\max_{t_{cur}}(f_s)$ is the estimated event onset up to t_{cur}. Firstly, it is easy to show the strictly monotonicity property of $\max_{t_{cur}}(f_s)$, i.e. $\max_{t_{cur}}(f_s) < \max_{t_{cur}+1}(f_s)$, on event duration as below

$$\begin{aligned}
\max_{t_{cur}}(f_s) &= \max\Big(\sum_{t'=t_0}^{t_{cur}} p_{t_s}(t|\mathbf{x}_{t'})\Big) \\
&< \max\Big(\sum_{t'=t_0}^{t_{cur}} p_{t_s}(t|\mathbf{x}_{t'}) + p_{t_s}(t|\mathbf{x}_{t_{cur}+1})\Big) \\
&= \max\Big(\sum_{t'=t_0}^{t_{cur}+1} p_{t_s}(t|\mathbf{x}_{t'})\Big) \\
&= \max_{t_{cur}+1}(f_s).
\end{aligned} \tag{23}$$

Inspecting in different disjoint segments of $[t_0, L)$, we have

$$\max_{t_{cur}}(f_s) = \max_{t_{cur}+1}(f_s) = 0, t_0 \leq t_{cur} < t_1, \tag{24}$$

$$\max_{t_{cur}}(f_s) < \max_{t_{cur}+1}(f_s), t_1 \leq t_{cur} \leq t_2, \tag{25}$$

$$\max_{t_{cur}}(f_s) = \max_{t_{cur}+1}(f_s) = \max_{t_e}(f_s), t_2 < t_{cur} < L. \tag{26}$$

The proof for the offset scoring function f_e can be obtained likewise. It can be interpreted that the more we know about the target event, the more we gain in terms of our prediction confidence.

Now, the question is how many superframes we need to detect an event reliably? To address this question, we need to determine a threshold for the confidence scores. For simplicity, we employ a common threshold for both onset and offset confidence scores f_s and f_e. Due to the fact that the scores are noisy, yet their peaks are dominant above the noise floor [11], the threshold just needs to be right above the noise floor for reliable detection. As soon as both accumulating scores reach the threshold, the event is considered detected as illustrated in Fig. 4. We determine the threshold by cross validation as described in Section 4.

3.2. Handling multi-class event categories

Our regression forests are specific for a target event category. In general, it is common that multiple event types are targeted. Out of $\mathcal{Y}$ event categories of interest, we trained a regression forest $\mathcal{F}^y$ for each category $y \in \{1, \ldots, \mathcal{Y}\}$. Due to the fact that the regression forests were trained with class-specific data, it is necessary to provide them with class-specific data to make proper estimates. We

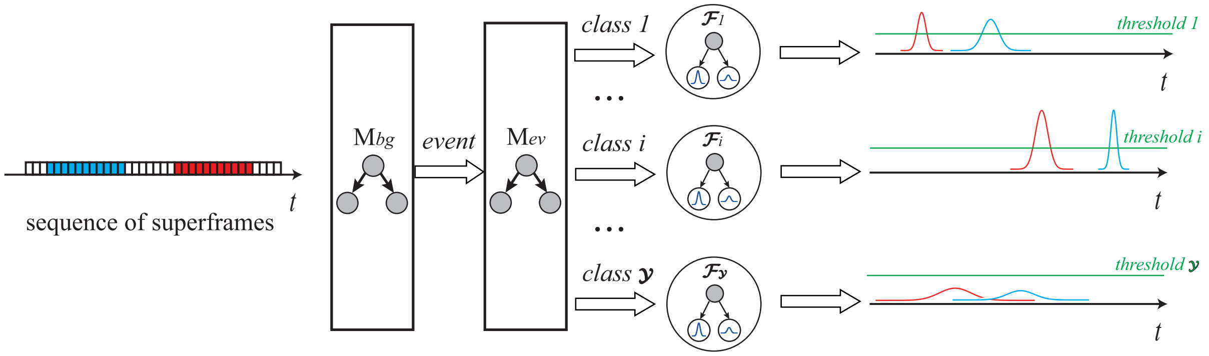


Fig. 5. The multi-class audio event detection system.

perform a superframe-wise classification step before regression. The architecture of the multi-class event detection system is illustrated in Fig. 5. The superframes are firstly passed to the binary classifier M_{bg} which filters out the background and only allows event superframes passing through. Subsequently, these event superframes are classified as one of the event categories of interest by the multi-class classifier M_{ev}. Finally, a superframe recognized as class y is inputted to the regression forest $\mathcal{F}^y$ for estimation.

We trained the classifiers M_{bg} and M_{ev} with random forest classification [13] which is computationally efficient for large data sets. More importantly, we take advantage of its probability support to further improve the system performance. An event superframe is weighted by its recognition confidence outputed by M_{ev}. As a result, (18) and (19) are re-written as:

$$f_s(t) = \sum_{t'} \delta_{\hat{y}_{t'},y} \cdot p(\hat{y}_{t'} = y) \cdot p_{t_s}(t|\mathbf{x}_{t'}), \tag{27}$$

$$f_e(t) = \sum_{t'} \delta_{\hat{y}_{t'},y} \cdot p(\hat{y}_{t'} = y) \cdot p_{t_e}(t|\mathbf{x}_{t'}). \tag{28}$$

Here, $\hat{y}_{t'}$ denotes the predicted label of $\mathbf{x}_{t'}$ and δ is Kronecker delta function. $p(\hat{y}_{t'} = y)$ is the probability that the predicted class label $\hat{y}_{t'}$ equals y. By weighting, a superframe recognized with higher confidence will reasonably contribute more into the estimation.

3.3. Event detection in action

Early event detection requires realtime processing, and therefore, target events, if they occur more than once, must be detected sequentially. We propose a detection mechanism as follows. The detector keeps two confidence scoring sequences, one for onset estimation and the other for offset estimation, centered at the current time index. Due to high variation in event durations, we set the size of the scoring sequences to twice the maximum length of the training events. The detector reads from a data stream and continuously monitors the occurrence of a target event. As a superframe arrives, its estimation is accumulated to the confidence scores. As long as we find a pair of maximum confidence scores above the detection threshold in chronological order, the maximum onset score at a position in the past and the maximum offset score in the future relative to the current time index, a target event is considered detected. Moreover, if a target event is detected, its temporal extent is determined as the interval starting from the estimated onset position and ending at the estimated offset position. Furthermore, due to the monotonicity of the scoring functions, during the event interval, both scoring peaks remain above the threshold. This provides an automatic mechanism to track the event. The event is recognized as complete when the position with the offset scoring peaks passes the current time index. After that, the process restarts to detect the upcoming target event. Thus, at any time, the detector needs to detect at most one target event.

4. EXPERIMENTS

4.1. Experiment setup

Databases. We conduct experiments on the ITC-Irst database [14] which was recorded in meeting-room environments and do not contain event overlap. The database consists of twelve recording sessions. There are totally 16 semantic event categories with approximately 50 events recorded for most of the categories. In accordance with the CLEAR 2006 challenge [15], we only took into account twelve classes for evaluation: door knock (kn), door slam (ds), steps (st), chair moving (cm), spoon cup jingle (cl), paper wrapping (pw), key jingle (kj), keyboard typing (kt), phone ring (pr), applause (ap), cough (co), laugh (la). The rest was considered as background. Many of the events are subtle (e.g. steps, chair moving, and keyboard typing), making the task more challenging. We used nine recording sessions for training and three remaining sessions for testing. Only one channel named *TABLE_1* [14] was used.

Features. The audio signals are downsampled to 16 kHz and decomposed into interleaved 100 ms long superframes with an overlap of 90 ms. The dense overlap is to ensure a high level of data correlation. The event superframes are labelled with the corresponding event labels and associated with the displacement vector to the event onset and offset. The background superframes are labelled with the class label 0, and no offset vectors are required.

To represent a superframe, we divide it into small 30 ms frames with Hamming window and 20 ms overlap. We utilize the set of 60 acoustic features suggested by Temko *et al.* [15] to represent a small frame. They consist of: (1) 16 log-frequency filter bank parameters, along with the first and second time derivatives, and (2) the following set of features: zero-crossing rate, short time energy, four sub-band energies, spectral flux calculated for each sub-band, spectral centroid, and spectral bandwidth. In turn, the superframe consisting of multiple small frames is represented by the empirical mean and the standard deviation of the frame feature vectors.

Parameters. To train random-forest classifiers M_{bg} and M_{ev}, we conservatively set the number of trees to 300 for both. The regression forests were trained with the random forest regression algorithm from Section 2 with ten random trees each. For a category y, a randomly sampled subset containing 50% superframes of the

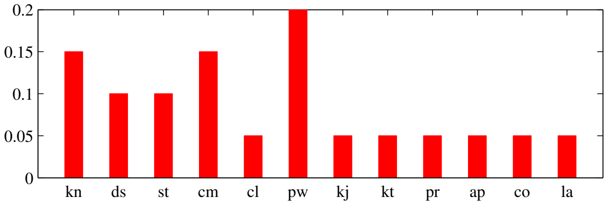


Fig. 6. Class-specific detection thresholds obtained by cross validation.

corresponding training set was used to train each random regression tree of $\mathcal{F}_y$. During training, 20,000 binary tests were generated for a split node. In addition, we set the maximum depth to $D_{\max} = 12$ and minimum number of superframes at leaf nodes to $N_{\min} = 10$.

Evaluation metrics. We used three metrics described in [11] for evaluation: *Acoustic Event Error Rate (AEER)*, *AED-ACC*, and *AED-ER*. The AEER and AED-ACC focus on the detection of event instances whereas AED-ER focuses more on AE localization where a good temporal alignment of the detected events is important. Note that AEER and AED-ER may exceed 100%. Further details of the metrics can be found in [11].

Baseline systems. We implemented two baseline systems similar to the *UPC-D* and *ITC-D1* systems in CLEAR 2006 evaluation [15]:

- **SVM**: This system is based on detection-by-classification with SVM classification. The setting is similar to that of the *UPC-D* system with one exception: an event hypotheses is rejected if its length is less than the minimum length of training events instead of the average length. This small change results in a significant performance improvement.
- **HMM**: This system is based on based on automatic speech recognition (ASR) framework. The employed features and parameters are the same as for the *ITC-D1* system. However, the data channel we used here is different from the original system.

4.2. Experimental results

In order to determine the detection threshold for each target event class, using nine training audio recordings, we conducted 9-fold leave-one-recording-out cross validation. The confidence scores were normalized to [0,1]. The threshold search was then performed on [0,1] with a step size of 0.05. Eventually, the threshold which yields maximum AED-ACC was adopted. During cross validation, we notice that it is unnecessary to train the regression forests but employed the one trained with the whole training data. It turned out that we only need to train the cross-validating classifiers M_{bg} and M_{ev}, which helps to significantly accelerate the cross validation process. As different event classes vary in their noise-floor characteristics [11], the thresholds are expected to be different. Fig. 6 shows the detection threshold for all target event classes.

Since the proposed algorithm is random, we conducted training and testing 10 times and report the average performance. However, the threshold search was performed only once in the first run and the found thresholds were re-used for the subsequent runs.

The accuracies of the the classifiers M_{bg} and M_{ev} are reported on superframe-wise basis. With independent testing, their accuracies are 86.99 ± 0.03% and 74.5 ± 0.06%, respectively. If we test them sequentially, M_{bg} followed by M_{ev}, the accuracy of the M_{ev} declines to 70.05 ± 0.07% since the wrong classification of M_{bg} is transferred to the next step. It should be emphasized here that only 70.05%, on average, of the truth event superframes which were correctly recognized really went into the regression forests subsequently to make estimation. Furthermore, the estimation is also perished by averagely 29.32% of false-positive superframes.

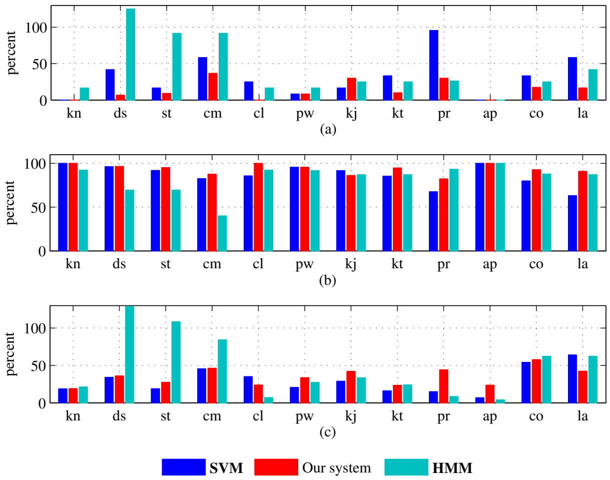


Fig. 7. Event detection results on different event classes: (a) AEER, (b) AED-ACC, and (c) AED-ER.

Table 1. AED performance comparison with the baseline systems on ITC-Irst database.

	Our systems	**SVM**	**HMM**
AEER (%)	**14.93 ± 0.95**	30.82	39.04
AED-ACC (%)	**92.43 ± 0.40**	83.73	84.35
AED-ER (%)	36.20 ± 1.49	**28.41**	36.71

For the detection task, we report the results for both offline and online running and we show that the online system offers the same performance as the offline system, but the events are detected much earlier and before they are finished. The overall detection results of the offline system are shown in Table 1. These results are slightly better than those reported in [11] which can be explained by the class-specific detection thresholds opposed to their common threshold for all target event categories. Compared to the baseline systems, our system roughly outperforms **SVM** by 16% on AEER and outruns **HMM** by 8% on AED-ACC. The detection results on different target event classes are illustrated in Fig. 7.

For online testing, we simulated the event data as sequential arrival. As a new event superframe arrives, we evaluate and record the performance. In Fig. 8, we show how the online detection performance develops as function of the number of observed event superframes. We also plot the offline performance as the baseline. Overall, as more superframes are observed, AED-ACC and AED-ER get better until they reach the offline baselines. It can also be seen that the online AED-ACC curves always reach the offline AED-ACC baselines before the maximum length of the events. That is, the events that can be detected correctly by the system are always detected before they finish, although the earliness varies for different event types. For example, from the AED-ACC curve of the class "ap", about 50% of events are correctly detected when approximately 75 superframes (about 0.75 second) are seen. After that,

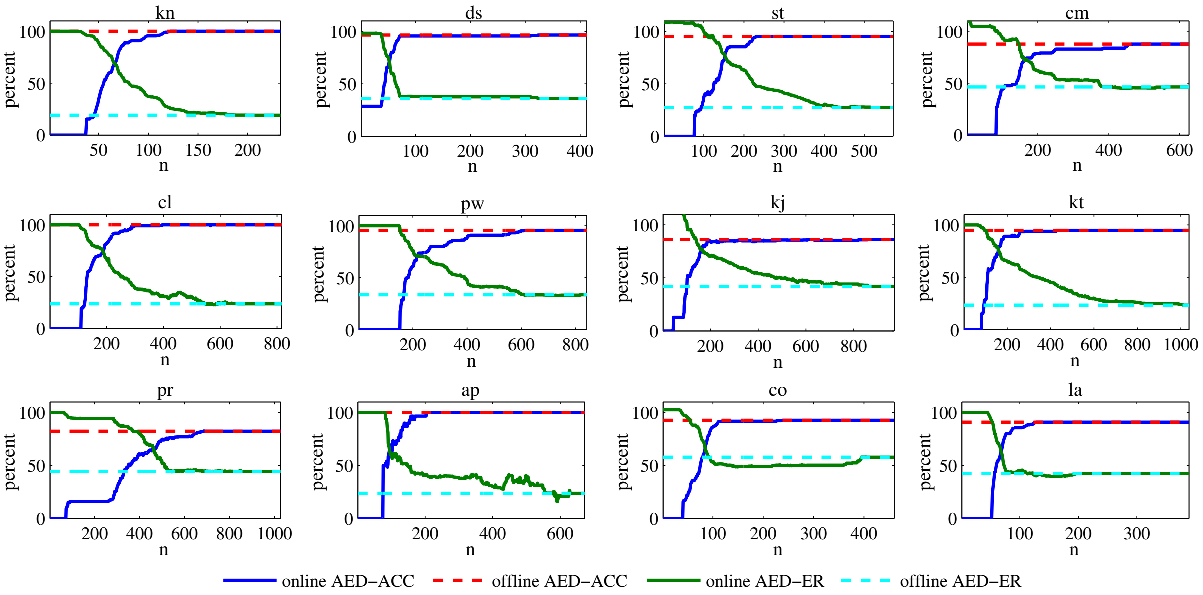


Fig. 8. Online event detection results on different event classes as functions of the number of observed superframes.

the curve goes upward until reaching the offline AED-ACC baseline off 100% after observing about 200 superframes (equivalent to 2 seconds). Considering that the "ap" events last more than 600 superframes, the online system only needs 30% of the event intervals to achieve the same detection accuracy as the offline system.

5. CONCLUSIONS

We presented an efficient early audio event detection system for audio streams based on random regression forest framework. Since the system only needs a partial event to make a decision, it can rapidly, yet reliably, detect the events coming in sequential order. In the experiments on the real-world recordings of the ITC-Irst database, we showed that the proposed system can detect the events of all target event categories at their very early stage without sacrificing the detection accuracy compared to the offline system which used the complete events. In addition, it also remains superior compared to the common approaches which cannot easily be adapted to perform reliable early detection.

6. REFERENCES

[1] C. Canton-Ferrer, T. Butko, C. Segura, X. Giro, C. Nadeu, J. Hernando, and J. R. Casas, "Audiovisual event detection towards scene understanding," in *Proc. CVPR Workshops*, 2009, pp. 81–88.

[2] A. Temko and C. Nadeu, "Acoustic event detection in meeting-room environments," *Pattern Recognition Letters*, vol. 30, pp. 1281–1288, 2009.

[3] P. K. Atrey, N. C. Maddage, and M. S. Kankanhalli, "Audio based event detection for multimedia surveillance," in *Proc. ICASSP*, 2006, pp. 813–816.

[4] V. Q. Nguyen, H. Kang, S.-T. Chung, S. Cho, K. Lee, and T. Seol, "Real-time audio surveillance system for PTZ camera," in *Proc. International Conference on Advanced Technologies for Communications (ATC)*, 2013, pp. 2162–1020.

[5] M. Hoai and F. De la Torre, "Max-margin early event detectors," in *Proc. CVPR*, 2012, pp. 2863–2870.

[6] D. Huang, S. Yao, Y. Wang, and F. De La Torre, "Sequential max-margin event detectors," in *Proc. ECCV*, 2014, pp. 410–424.

[7] A. Plinge, R. Grzeszick, and G. Fink, "A bag-of-features approach to acoustic event detection," in *Proc. ICASSP*, 2014, pp. 3704–3708.

[8] S. Ikbal and T. Faruquie, "HMM based event detection in audio conversation," in *Proc. ICME*, 2008, pp. 1497–1500.

[9] X. Zhuang, X. Zhou, M. A. Hasegawa-Johnson, and T. S. Huang, "Real-world acoustic event detection," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1543–1551, 2010.

[10] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning," *Foundations and Trends in Computer Graphics and Vision*, vol. 7, no. 2–3, pp. 81–227, 2012.

[11] H. Phan, M. Maaß, R. Mazur, and A. Mertins, "Random regression forests for acoustic event detection and classification," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 23, no. 1, pp. 20–31, 2015.

[12] H. Phan, M. Maaß, R. Mazur, and A. Mertins, "Acoustic event detection and localization with regression forests," in *Proc. Interspeech*, 2014, pp. 2524–2528.

[13] L. Breiman, "Random forest," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[14] C. Zieger and M. Omologo, "Acoustic event detection - ITC-Irst AED database," Tech. Rep., Internal ITC report, 2005.

[15] A. Temko, R. Malkin, C. Zieger, D. Macho, C. Nadeu, and M. Omologo, "CLEAR evaluation of acoustic event detection and classification systems," *Lecture Notes in Computer Science*, vol. 4122, pp. 311–322, 2007.