Robust Endpoint Detection and Energy Normalization for Real-Time Speech and Speaker Recognition

Qi Li, *Senior Member, IEEE*, Jinsong Zheng, Augustine Tsai, and Qiru Zhou, *Member, IEEE*

***Abstract*—When automatic speech recognition (ASR) and speaker verification (SV) are applied in adverse acoustic environments, endpoint detection and energy normalization can be crucial to the functioning of both systems. In low signal-to-noise ratio (SNR) and nonstationary environments, conventional approaches to endpoint detection and energy normalization often fail and ASR performances usually degrade dramatically. The purpose of this paper is to address the endpoint problem. For ASR, we propose a real-time approach. It uses an optimal filter plus a three-state transition diagram for endpoint detection. The filter is designed utilizing several criteria to ensure accuracy and robustness. It has almost invariant response at various background noise levels. The detected endpoints are then applied to energy normalization sequentially. Evaluation results show that the proposed algorithm significantly reduces the string error rates in low SNR situations. The reduction rates even exceed 50% in several evaluated databases. For SV, we propose a batch-mode approach. It uses the optimal filter plus a two-mixture energy model for endpoint detection. The experiments show that the batch-mode algorithm can detect endpoints as accurately as using HMM forced alignment while the proposed one has much less computational complexity.**

***Index Terms*—Change-point detection, edge detection, endpoint detection, optimal filter, robust speech recognition, speaker verification, speech activity detection, speech detection.**

I. INTRODUCTION

IN SPEECH and speaker recognition, we need to process the signal in utterances consisting of speech, silence, and other background noise. The detection of the presence of speech embedded in various types of nonspeech events and background noise is called *endpoint detection*, speech detection, or speech activity detection. In this paper, we address endpoint detection by sequential and batch-mode processes to support real-time recognition (in which the recognition response is the same as or faster than recording an utterance). The sequential process is often used in automatic speech recognition (ASR) [1] while the batch-mode process is often allowed in speaker recognition [2], name dialing [3], command control and embedded systems, where utterances are usually as short as a few seconds and the delay in response is usually small.

Endpoint detection has been studied for several decades. The first application was in a telephone transmission and switching system developed in Bell Labs, for time assignment of communication channels [4]. The principle was to use the free channel time to interpolate additional speakers by speech activity detection. Since then, various speech detection algorithms have been developed for ASR, speaker verification, echo cancellation, speech coding and other applications. In general, different applications need different algorithms to meet their specific requirements in terms of computational accuracy, complexity, robustness, sensitivity, response time, etc. The approaches include those based on energy threshold (e.g., [5]), pitch detection (e.g., [6]), spectrum analysis, cepstral analysis [7], zero-crossing rate [8], [9], periodicity measure, hybrid detection [10], fusion [11] and many other methods. Furthermore, similar issues have also been studied in other research areas, such as edge detection in image processing [12], [13] and change-point detection in theoretical statistics [14]–[18].

As is well-known, endpoint detection is crucial to both ASR and speaker recognition because it often affects a system's performance in terms of accuracy and speed for several reasons. First, cepstral mean subtraction (CMS) [19]–[21], a popular algorithm for robust speaker and speech recognition, needs accurate endpoints to compute the mean of speech frames precisely in order to improve recognition accuracy. Second, if silence frames can be removed prior to recognition, the accumulated utterance likelihood scores will focus more on the speech portion of an utterance instead of on both noise and speech. Therefore, it has the potential to increase recognition accuracy. Third, it is hard to model noise and silence accurately in changing environments. This effect can be limited by removing background noise frames in advance. Fourth, removing nonspeech frames when the number of nonspeech frames is large can significantly reduce the computation time. Finally, for open speech recognition systems, such as open-microphone desktop applications and audio transcription of broadcast news, it is necessary to segment utterances from continuous audio input.

In applications of speech and speaker recognition, nonspeech events and background noise complicate the endpoint detection problem considerably. For example, the endpoints of speech are often obscured by speaker-generated artifacts such as clicks, pops, heavy breathing, or by dial tones. Long-distance telephone transmission channels also introduce similar types of artifacts and background noise. In recent years, as wireless, hands-free and Internet Protocol (IP) phones get more and more popular, the endpoint detection problem becomes even more difficult since the signal-to-noise ratios (SNR) of these kinds of communication devices are usually lower and the noise is nonstationary than those in traditional telephone lines and handsets. The noise may come from the background, such as car noise, room reflec-

Manuscript received June 7, 2001; revised February 13, 2002. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Juergen Schroeter.

The authors are with the Multimedia Communications Research Lab, Bell Labs, Lucent Technologies, Murray Hill, NJ 07974 USA (e-mail: qli@research.bell-labs.com).

Publisher Item Identifier S 1063-6676(02)03972-X.

tion, street noise, background talking, etc., or from communication systems, such as coding, transmission, packet loss, etc. In these cases, the ASR or speaker recognition performance often degrades dramatically due to unreliable endpoint detection.

Another problem related to endpoint detection is real-time energy normalization. In both ASR and speaker recognition, we usually normalize the energy feature such that the largest energy level in a given utterance is close to or slightly below a constant of zero or one. This is not a problem in batch-mode processing, but it can be a crucial problem in real-time processing since it is difficult to estimate the maximal energy in an utterance with just a short-time data buffer while the acoustic environment is changing. It becomes especially hard in adverse acoustic environments. A look-ahead approach to energy normalization can be found in [6]. Actually, as we will point out later in this study, real-time energy normalization and endpoint detection are two related problems. The more accurately we can detect endpoints, the better we can do on real-time energy normalization.

In this paper, we propose two endpoint-detection algorithms for real-time ASR and speaker recognition. Generally speaking, both algorithms must meet the following requirements: accurate location of detected endpoints; robust detection at various noise levels; low computational complexity; fast response time; and simple implementation. The real-time energy normalization problem is addressed together with endpoint detection.

The rest of the paper is organized as follows. In Section II, we will introduce a filter for endpoint detection. In Section III, we will propose a sequential algorithm of combined endpoint detection and energy normalization for ASR in adverse environments and provide experimental results in large database evaluations. In Section IV, we will propose an accurate endpoint-detection algorithm for batch-mode applications and compare the detected endpoints with manually-detected as well as HMM forced-alignment detected endpoints. Finally, we will summarize our findings in Section V.

II. A Filter for Endpoint Detection

To ensure the low-complexity requirement, we borrow the one-dimensional (1-D) short-term energy in the cepstral feature to be the feature for endpoint detection

$$g(t) = 10\log_{10} \sum_{j=n_t}^{n_t+I-1} o(j)^2 \tag{1}$$

where

$o(j)$	data sample;
t	frame number;
$g(t)$	frame energy in decibels;
I	window length;
n_t	number of the first data sample in the window.

Thus, the detected endpoints can be aligned to the ASR feature vector automatically and the computation can be reduced from the speech-sampling rate to the frame rate.

For accurate and robust endpoint detection, we need a detector that can detect all possible endpoints from the energy feature. Since the output of the detector contains false acceptances, a decision module is then needed to make final decisions based on the detector's output.

Here, we assume that one utterance may have several speech segments separated by possible pauses. Each of the segments can be determined by detecting a pair of *endpoints* named segment *beginning* and *ending points*. On the energy contours of utterances, there is always a raising edge following a beginning point and a descending edge preceding an ending point. We call them *beginning* and *ending edges*, respectively, as shown in Fig. 4(a). Since endpoints always come with the edges, our approach is first to detect the edges and then to find the corresponding endpoints.

The foundation of the theory of the optimal edge detector was first established by Canny [12]. He derived an optimal step-edge detector. Spacek [22], on the other hand formed a performance measure combining all three quantities derived by Canny and provided the solution of the optimal filter for step edge. Petrou and Kittler then extended the work to ramp-edge detection [13]. Since the edges corresponding to endpoints in the energy feature are closer to the ramp edge than the ideal step edge, Li and Tsai applied Petrou and Kittler's filter to the endpoint detection for speaker verification in [2].

In summary, we need a detector that meets the following general requirements:

1) invariant outputs at various background energy levels;
2) capability of detecting both beginning and ending points;
3) short time delay or look-ahead;
4) limited response level;
5) maximum output signal-to-noise ratio (SNR) at endpoints;
6) accurate location of detected endpoints;
7) maximum suppression of false detection.

We then need to convert the above criteria to a mathematic representation. As we have discussed, it is reasonable to assume that the beginning edge in the energy contour is a ramp edge that can be modeled by the following function:

$$c(x) = \begin{cases} 1 - \frac{e^{-sx}}{2}, & \text{for } x \geq 0 \\ \frac{e^{sx}}{2}, & \text{for } x \leq 0 \end{cases} \tag{2}$$

where x represents the frame number of the feature and s is some positive constant which can be adjusted for different kinds of edges, such as beginning or ending edges and for different sampling rates.

The detector is a 1-D filter $f(x)$ which can be operated as a moving-average filter in the energy feature. From the above requirements, the filter should have the following properties which are similar to those in [13].

P1) It must be antisymmetrical, i.e., $f(x) = -f(-x)$ and thus $f(0) = 0$. This follows from the fact that we want it to detect an antisymmetrical features [12], i.e., sensitive to both beginning and ending edges according to the request in 2); and to have near-zero response to background noise at any level, i.e., invariant to background noise according to the request in 1).

P2) According to the requirement in 3), it must be of finite extent going smoothly to zero at its ends: $f(\pm w) = 0$,

$f'(\pm w) = 0$ and $f(x) = 0$ for $|x| \geq w$, where w is the half width of the filter.

P3) According to the requirement in 4), it must have a given maximum amplitude $|k|$: $f(x_m) = k$ where x_m is defined by $f'(x_m) = 0$ and x_m is in the interval $(-w, 0)$.

If we further represent requirements 5), 6), and 7), as $S(f(x))$, $L(f(x))$ and $C(f(x))$, respectively, the combined objective function is

$$J = \max_{f(x)} F\{S(f(x)), L(f(x)), C(f(x))\}; \tag{3}$$

Subject to properties P1), P2), and P3).

It aims at finding the filter function, $f(x)$, such that the value of the objective function F is maximal subject to properties P1)–P3). Fortunately, the object function is very similar to optimal edge detection in image processing and the details of the object function have been derived by Petrou and Kittler [13] following Canny [12], as well as in Appendix A.

After applying the method of Lagrange multipliers, the solution for the filter function is [13]

$$\begin{aligned} f(x) =& e^{Ax}\left[K_1 \sin(Ax) + K_2 \cos(Ax)\right] \\ &+ e^{-Ax}\left[K_3 \sin(Ax) + K_4 \cos(Ax)\right] \\ &+ K_5 + K_6 e^{sx} \end{aligned} \tag{4}$$

where A and K_i are filter parameters. Since $f(x)$ is only half of the filter, when $w = W$, the actual filter coefficients are

$$h(i) = \{-f(-W \leq i \leq 0),\ f(1 \leq i \leq W)\} \tag{5}$$

where i is an integer. The filter can then be operated as a moving-average filter in

$$F(t) = \sum_{i=-W}^{W} h(i) g(t+i) \tag{6}$$

where $g(\cdot)$ is the energy feature and t is the current frame number. An example of the designed optimal filter is shown in Fig. 1. Intuitively, the shape of the filter indicates that the filter must have positive response to a beginning edge, negative response to an ending edge and a near zero response to silence. Its response is basically invariant to different background noise levels since they all have near zero responses.

III. Real-Time Endpoint Detection and Energy Normalization for ASR

The approach of using endpoint detection for real-time ASR is illustrated in Fig. 2 [23]. We use an optimal filter, as discussed in the last section, to detect all possible endpoints, following by a three-state logic as a decision module to decide real endpoints. The information of detected endpoints is also utilized for real-time energy normalization. Finally, all silence frames are removed and only the speech frames including cepstrum and the normalized energy are sent to the recognizer.

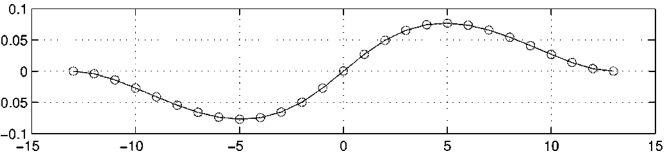

Fig. 1. Shape of the designed optimal filter.

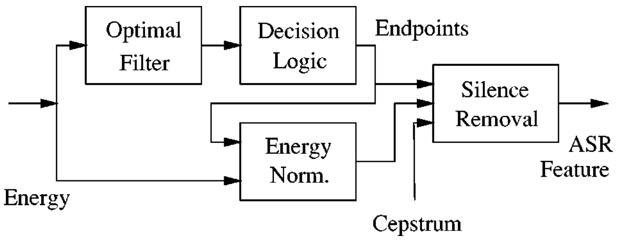

Fig. 2. Endpoint detection and energy normalization for real-time ASR.

A. Filter for Both Beginning- and Ending-Edge Detection

After evaluating the shapes of both beginning and ending edges, we choose the filter size to be $W = 13$ to meet requirements 2) and 3).

For $W = 7$ and $s = 1$, the filter parameters have been provided in [13] as $A = 0.41$, $[K_1 \ldots K_6] = [1.583, 1.468, -0.078, -0.036, -0.872, -0.56]$. For $W = 13$ in our application, we just need to rescale $s = 7/W = 0.5385$ and $A = 0.41s = 0.2208$ while K_i's are as shown previously.

The shape of the designed filter is shown in Fig. 1 with a simple normalization, $h/13$. For real-time detection, let $H(i) = h(i-13)$; then the filter has 25 points in total with a 24-frame look-ahead since both $H(1)$ and $H(25)$ are zeros. The filter operates as a moving-average filter

$$F(t) = \sum_{i=2}^{24} H(i) g(t+i-2) \tag{7}$$

where $g(\cdot)$ is the energy feature and t is the current frame number. The output $F(t)$ is then evaluated in a three-state transition diagram for final endpoint decisions.

B. State Transition Diagram

Endpoint decision needs to be made by comparing the value of $F(t)$ with some pre-determined thresholds. Due to the sequential nature of the detector and the complexity of the decision procedure, we use a three-state transition diagram to make final decisions.

As shown in Fig. 3, the three states are: *silence, in-speech,* and *leaving-speech.* Either the silence or the in-speech state can be a starting state and any state can be a final state. In the following discussion, we assume that the silence state is the starting state. The input is $F(t)$ and the output is the detected frame numbers of beginning and ending points. The transition conditions are labeled on the edges between states and the actions are listed in parentheses. "Count" is a frame counter, T_U and T_L are two thresholds with $T_U > T_L$ and "Gap" is an integer indicating the required number of frames from a detected endpoint to the actual end of speech.

We use Fig. 4 as an example to illustrate the state transition. The energy for a spoken digit "4" is plotted in Fig. 4(a) and the

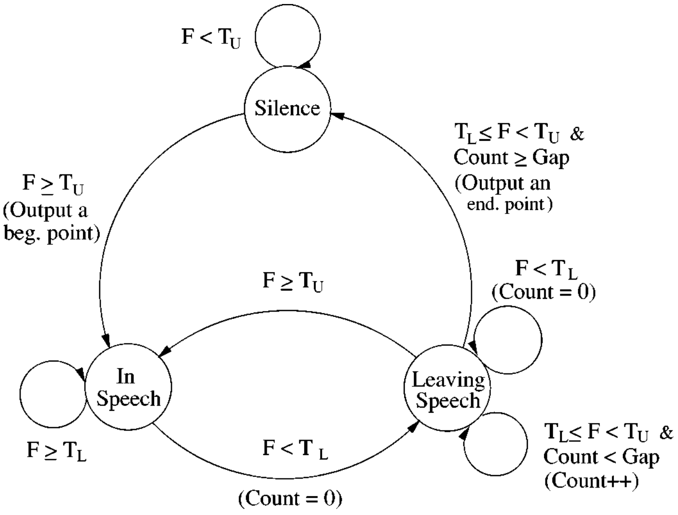

Fig. 3. State transition diagram for endpoint decision.

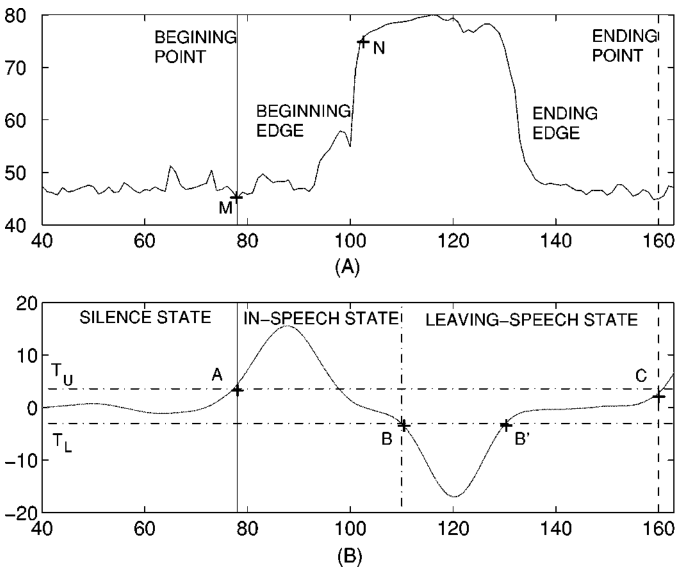

Fig. 4. Example: (a) energy contour of digit "4" and (b) filter outputs and state transitions.

filter output is shown in Fig. 4(b). The state diagram stays in the silence state until $F(t)$ reaches point A in Fig. 4(b), where $F(t) \geq T_U$ means that a beginning point is detected. The actions are to output a beginning point [corresponding to the left vertical solid line in Fig. 4(a)] and to move to the in-speech state. It stays in the in-speech state until reaching point B in Fig. 4(b), where $F(t) < T_L$. The diagram then moves to the leaving-speech state and sets Count $= 0$. The counter resets several times until reaching point B'. At point C, Counter $=$ Gap $= 30$. An actual endpoint is detected as the left vertical dashed line in Fig. 4(b). The diagram then moves back to the silence state. During the stay in the leaving-speech state, if $F(t) > T_U$, this means that a beginning edge is coming and we should move back to the in-speech state. The 30-frame gap corresponds to the period of descending energy before reaching a real ending point.

We note that the thresholds, such as T_U and T_L, are set in the filter outputs instead of absolute energy. Since the filter output is stable to the noise levels, the detected endpoints are more reliable. Those constants, Gap, T_U, and T_L, can be determined empirically by plotting several utterances and corresponding filter outputs. As we will show in the database evaluation, the algorithm is not very sensitive to the values of T_U and T_L since the same values were used in different databases. Also, in some applications, two separate filters can be designed for beginning and ending point detection. The size of the beginning filter can be smaller than 25 points while the ending filter can be larger than 25 points. This approach may further improve accuracy; however, it will have a longer delay and use more computation. The 25-point filter used in this section was designed for both beginning and ending point detection in an 8 KHz sampling rate. Also, in the case that an utterance is started from an unvoiced phoneme, it is practical to step back about ten frames from the detected beginning points.

C. Real-Time Energy Normalization

Suppose that the maximal energy value in an utterance is $g_{\max}$. The purpose of energy normalization is to normalize the utterance energy $g(t)$, such that the largest value of energy is close to zero by performing $\tilde{g}(t) = g(t) - g_{\max}$. In a real-time mode, we have to estimate the maximal energy $g_{\max}$ sequentially while the data are being collected. Here, the estimated maximum energy becomes a variable and is denoted as $\hat{g}_{\max}(t)$. Nevertheless, we can use the detected endpoints to obtain a better estimate.

We first initialize the maximal energy to a constant g_0, which is selected empirically and use it for normalization until we detect the first beginning point at M as in Fig. 4, i.e., $\hat{g}_{\max}(t) = g_0, \forall t < M$. If the average energy

$$\bar{g}(t) = E\{g(t); M \leq t \leq M + 2W\} \geq g_m \qquad (8)$$

where g_m is a pre-selected threshold to ensure that new $\hat{g}_{\max}$ is not from a single click, we then estimate the maximal energy as

$$\hat{g}_{\max}(t) = \max\{g(t); M \leq t \leq M + 2W\} \qquad (9)$$

where $2W+1 = 25$ is the length of the filter and $2W$ the length of the look-ahead window. At point M, the look-ahead window is from M to N as shown in Fig. 4. From now on, we update $\hat{g}_{\max}(t)$ as

$$\hat{g}_{\max}(t) = \max\{g(t + 2W), \hat{g}_{\max}(t - 1); \forall t > M\}. \qquad (10)$$

Parameter g_0 may need to be adjusted for different system. For example, the value of g_0 could be different between telephone and desktop systems. Parameter g_m is relatively easy to determine.

For the example in Fig. 5, the energy features of two utterances with 20 dB SNR (bottom) and 5 dB SNR (top) are plotted in Fig. 5(a). The 5-dB utterance is generated by artificially adding car noise to the 20 dB one. The filter outputs are shown in Fig. 5(b) for 20 dB (solid line) and 5 dB (dashed line) SNRs, respectively. The detected endpoints and normalized energy for 20 and 5 dB SNRs are plotted in Fig. 5(c) and 5(d), respectively. We note that the filter outputs for 20 and 5 dB cases

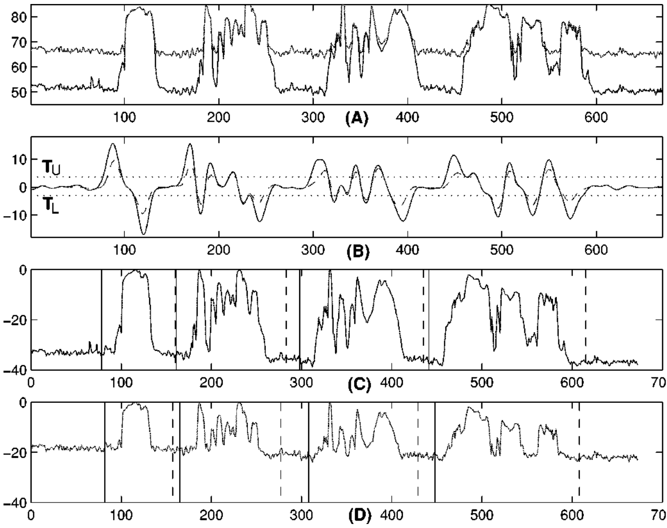

Fig. 5. (a) Energy contours of "4-327-631-Z214" from original utterance (bottom, 20 dB SNR) and after adding car noise (top, 5 dB SNR). (b) Filter outputs for 5 dB (dashed line) and 20 dB (solid line) SNR cases. (c) Detected endpoints and normalized energy for the 20 dB SNR case and (d) for the 5 dB SNR case.

are almost invariant around T_L and T_U, although their background energy levels have a difference of 15 dB. This ensures the robustness in endpoint detection. We also note that the normalized energy profiles are almost the same as the original one, although the normalization is done in a real-time mode.

D. Database Evaluation

The proposed algorithm was compared with a baseline endpoint detection algorithm on one noisy database and several telephone databases.

1) Baseline Endpoint Detection: The baseline system is a real-time, energy contour based adaptive detector developed based on the algorithm introduced in [1], [5]. It has been used for years in research and commercial speech recognizers. In the baseline system, a six-state transition diagram is used to detect endpoints. Those states are named as *initializing*, *silence*, *rising*, *energy*, *fell-rising*, and *fell* states. In total, eight counters and 24 hard-limit thresholds are used for the decisions of state transition. Two adaptive threshold values were used in most of the thresholds. We note that all the thresholds are compared with raw energy values directly.

Energy normalization in the baseline system is done separately by estimating the maximal and minimal energy values, then comparing their difference to a fixed threshold for decision. Since the energy values change with acoustic environments, the baseline approach leads to unreliable endpoint detection and energy normalization, especially in low SNR and nonstationary environments.

2) Noisy Database Evaluation: In this experiment, a database was first recorded from a desktop computer at 16 KHz sampling rate, then down-sampled to 8 KHz sampling rate. Later, car and other background noises were artificially added to the original database at the SNR levels of 5, 10, 15, and 20 dB. The original database has 39 utterances and 1738 digits in total. Each utterance has 3, 7, or 11 digits. LPC feature and the short-term energy were used and the hidden Markov model (HMM) in a head-body-tail (HBT) structure was employed to model each of the digits [24], [25]. The HBT structure assumes that context dependent digit models can be built by concatenating a left-context dependent unit (head) with a context independent unit (body) followed by a right-context dependent unit (tail). We used three HMM states to represent each "head" and "tail" and four state to represent each "body." Sixteen mixtures were used for each body state and four mixtures were used for each head or tail state.

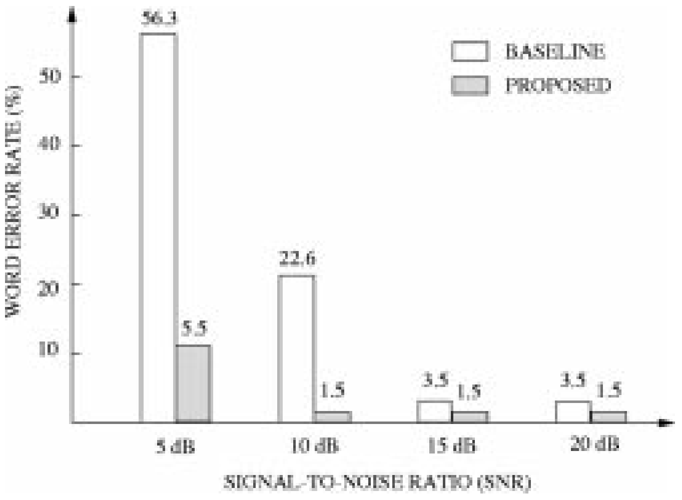

Fig. 6. Comparisons on real-time connected digit recognition with various SNRs. From 5- to 20-dB SNRs, the proposed algorithm provided word error rate reductions of 90.2%, 93.4%, 57.1%, and 57.1%, respectively.

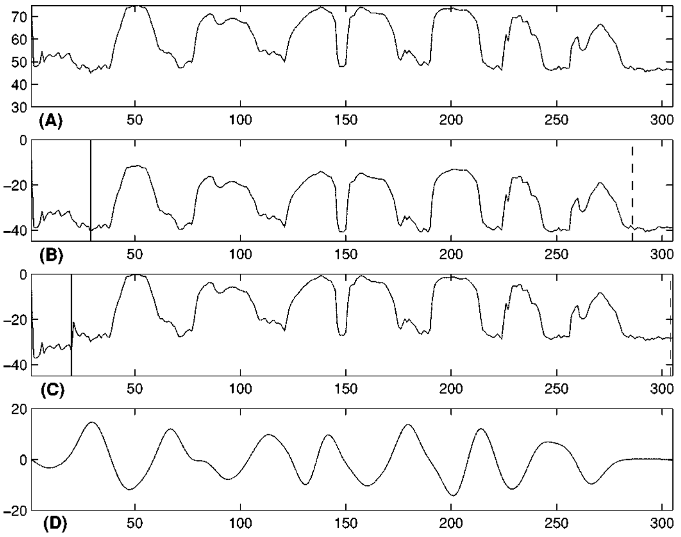

Fig. 7. (a) Energy contour of the 523th utterance in DB5: "1 Z 4 O 5 8 2." (b) Endpoints and normalized energy from the baseline system. The utterance was recognized as "1 Z 4 O 5 8." (c) Endpoints and normalized energy from the proposed system. The utterance was recognized correctly as "1 Z 4 O 5 8 2." (d) The filter output.

The real-time recognition performances on various SNRs are shown in Fig. 6. Compared to the baseline algorithm, the proposed one significantly reduced word error rates. The baseline algorithm failed to work in low SNR cases because it uses raw energy values directly to detect endpoints and to perform energy normalization. The proposed algorithm makes decision on the filter output instead of raw energy values; therefore, it provided more robust results. An example of error analysis is shown in Fig. 7.

TABLE I
DATABASE EVALUATION RESULTS (%)

Database IDs (Number of strings, Number of words)	Word Error Rate		Word Error Reduction
	Base- line	Pro- posed	
DB1 (232, 1393)	13.7	11.8	13.9
DB2 (671, 1341)	14.6	7.9	45.9
DB3 (1957,1957)	4.5	4.4	2.2
DB4 (272, 1379)	10.0	9.6	4.0
DB5 (259, 2632)	15.8	15.7	0.6
DB6 (576, 1738)	2.8	1.1	60.7
DB7 (583, 1743)	1.7	1.5	11.8
DB8 (664, 2087)	0.9	0.7	22.2
DB9 (619, 8194)	1.0	0.7	30.0
DB10 (651, 8452)	5.7	5.6	1.8
DB11 (707, 9426)	1.6	1.4	12.5

3) Telephone Database Evaluation: The proposed algorithm was further evaluated in 11 databases collected from the telephone networks with 8 kHz sampling rates in various acoustic environments. LPC parameters and short-term energy were used. The acoustic model consists of one silence model, 41 mono-phone models and 275 head-body-tail units for digit recognition. It has a total of 79 phoneme symbols, 33 of which are for digit units. Eleven databases, DB1 to DB11, were used for the evaluation. DB1 to DB5 contain digits, alphabet and word strings. Finite-state grammars were used to specify the valid forms of recognized strings. DB6 to DB11 contain pure digit strings. In all the evaluations, both endpoint detection and energy normalization were performed in real-time mode and only the detected speech portions of an utterance were sent to the recognition back-end.

In the proposed system, we set the parameters as $g_0 = 80.0$, $g_m = 60.0$, $T_U = 3.6$, $T_L = -3.0$ and Gap $= 30$. These parameters were unchanged throughout the evaluation in all 11 databases to show the robustness of the algorithm, although the parameters can be adjusted according to signal conditions in different applications. The evaluation results are listed in Table I. It shows that the proposed algorithm works very well in regular telephone data as well. It provided word error reduction in most of the databases. The word error reductions even exceed 30% in DB2, DB6, and DB9.

To analyze the improvement, the original energy feature of an utterance, "1 Z 4 O 5 8 2," in DB6 is plotted in Fig. 7(a). The detected endpoints and normalized energy using the conventional approach are shown in Fig. 7(b) while the results of the proposed algorithm are shown in Fig. 7(c). The filter output is plotted in Fig. 7(d). From Fig. 7(b), we can observe that the normalized maximal energy of the conventional approach is about 10 dB below zero, which causes a wrong recognition result: "1 Z 4 O 5 8." On the other hand, the proposed algorithm normalized the maximal energy to zero approximately and the utterance was recognized correctly as "1 Z 4 O 5 8 2."

IV. ACCURATE BATCH-MODE ENDPOINT DETECTION FOR SPEAKER VERIFICATION

So far, we have focused on real-time endpoint detection, which is mainly for ASR applications where silence or garbage models are usually used to further determine accurate endpoints in decoding. In another category of applications, real-time processing is not so crucial. Speech data can be processed in a batch-mode, i.e., after data recording is finished. The applications include speaker verification, name dialing, speech control, etc., where the utterances are usually short (e.g., less than 2 s) and the verification or recognition can be done within 1 s. Since many of these kinds of applications are offered in embedded systems, such as wireless phones or portable devices; or in multi-user systems, such as a speaker verification server for millions of users [3], they normally require low computational complexity for low cost or for a fast response. For these cases, one solution is to use an accurate end-point detector to remove all silence; therefore, we not only can reduce the number of decoding frames significantly, but also eliminate the silence model in decoding, which usually takes a lot of space and computation. Batch-mode processing enables this class of operations.

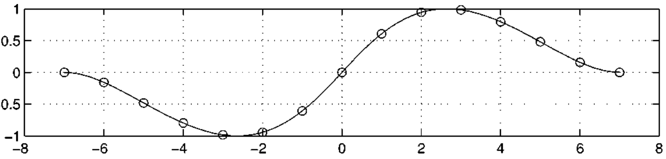

Fig. 8. Shape of the optimal filter for beginning edge detection, plotted as $h_b(t)$, with $W = 7$ and $s = 1$.

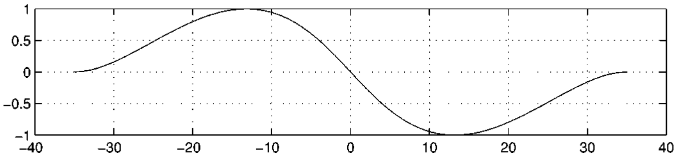

Fig. 9. Shape of the optimal filter for ending edge detection, plotted as $h_e(t)$, with $W = 35$ and $s = 0.2$.

A. Batch-Mode Algorithm

To obtain accurate endpoints, we designed two filters, one for beginning-edge and another for ending-edge detection, using the algorithm in Section II. The first filter is shown in Fig. 8 with seven points, while the second one is shown in Fig. 9 with 35 points. This is because the ending edge is usually longer than the beginning edge. We note that the ending filter gives positive response at a detected ending edge. To help in accurately determining energy thresholds, we use a Gaussian mixture model to model the energy distribution. The final endpoints are detected by combining the information from the filter outputs and the estimated thresholds.

1) Energy Distribution Model: We assume that a Gaussian mixture model can approximately represent the distribution of energy in an utterance with two mixtures representing speech and background energy, respectively

$$p(g;\mu_1,\mu_2,\sigma_1,\sigma_2,c) = c\mathcal{N}_1(g;\mu_1,\sigma_1) + (1-c)\mathcal{N}_2(g;\mu_2,\sigma_2) \tag{11}$$

where c is a weighting parameter, $\mathcal{N}_i$ is a normal distribution given by

$$\mathcal{N}_i(g;\mu_1,\sigma_1) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-(g-\mu_i)^2/(2\sigma_i^2)} \tag{12}$$

and μ_i and σ_i are the mean and standard deviation, respectively. The means for speech and background noise are $\mu_v = \max\{\mu_1, \mu_2\}$ and $\mu_n = \min\{\mu_1, \mu_2\}$ with the corresponding standard deviations, σ_v and σ_n. The thresholds for speech and background noise are $\theta_v = \mu_v - \sigma_v$ and $\theta_n = \mu_n + \sigma_n$, respectively. When the energy value is above θ_v, we consider it as speech; when the energy value is below θ_n, we consider it as background noise. To obtain fast and explicit parameter estimation, we applied a moment algorithm instead of the popular EM algorithm which needs iterations. The fast estimation algorithm is listed in Appendix B.

2) Summary of the Algorithm: We now summarize the proposed algorithm for batch-mode endpoint detection [2]. The parameters in the following algorithm are for the energy feature computed from 30 ms energy windows shifting every 10 ms. The data sampling rate is 8 KHz.

1) Compute log energy of the given utterance, $g(t)$ and normalize it by subtracting $\max_t\{g(t)\}$ to get $\tilde{g}(t)$. We assume that the speech is surrounded by silence and various kinds of noise.

2) Remove the dial tone from $\tilde{g}(t)$. The dial tone can be detected when

$$\tilde{g}(t) > -1.5,\ n \le t \le i,\ \text{and}\ i - n > 8.$$

These two parameters are determined based on the minimal length and minimal energy level of dial tones.

3) Estimate μ_v, μ_n, σ_v and σ_n using (22) to (29), then determine two thresholds

$$\theta_v = \mu_v - \sigma_v \text{ and } \theta_n = \mu_n + \sigma_n$$

for speech and background energy, respectively. Speech energy should be above the value of threshold θ_v and silence/background noise energy should be below θ_n. This is based on the assumption that noise and speech can be represented as two separated Gaussian mixtures.

4) Compute the output of the beginning-edge filter

$$y_b(t) = \sum_{i=-3}^{3} h_b(i) g(t+i-1) \tag{13}$$

then search for the locations of all peaks $R(k)$, from the filter output $y_b(t)$. A peak associated with a beginning point should meet the following properties:

$$\begin{aligned} y'(R(k)) &= 0, \\ y''(R(k)) &< 0, \text{ and} \\ y(R(k)) &> 0.2 \max\{y_b(t)\}. \end{aligned}$$

The actual beginning point is

$$B(k) = R(k) - 2$$

where k is the kth beginning edge. The shift is due to the offset between the center of a beginning edge and the actual beginning point.

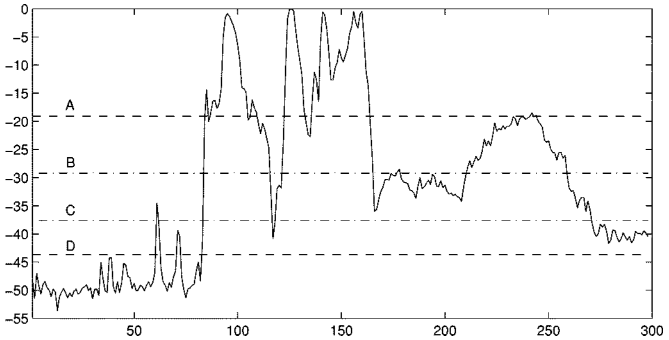

Fig. 10. Normalized log energy of "Call office" with heavy breath in the end. Lines A, B, C, and D indicate the estimated values of μ_v, θ_v, θ_n and μ_n, respectively.

5) From a detected beginning point $B(m)$, search for the corresponding ending point $E(m)$, which should satisfy the following conditions:

   i) $\tilde{g}(E(m)) \ge \theta_n$ and $\tilde{g}(E(m)+1) < \theta_n$;

   ii) $E(m) - B(m) \ge 6$;

   iii) when $B(m) \le t \le E(m)$, 60% frames of $g(t)$, should have the values above θ_v; and

   iv) $E(m) < B(m+1)$.

Here, ii) and iii) are to ensure that the segmentation is speech instead of a click or breath noise. The segment that cannot meet the above conditions is not a speech segment.

6) Determine the actual last ending point $E(m)$. Compute the response of the ending-edge filter in the last segment, $B(M) \le t \le E(M)$, by

$$y_e(t) = \sum_{i=-17}^{W=17} h_e(i) g(t+i-1). \tag{14}$$

Search for the last peak of $y_e(t)$, where $t = T$ and $y_e(T) \ge 0.6 \max\{y_e(t)\}$. Then, shift the peak point located at the center of the ending edge to the last ending point. The offset should be about half the filter size. We choose 16 frames. Thus, if at frame $T + 16$, the energy level is still higher than threshold θ_n, the ending point is at frame $T + 16$; otherwise, the ending point is the last point before the energy crosses the threshold θ_n

$$E(M) = \begin{cases} T + 16, & \text{if } \tilde{g}_e(T+16) \ge \theta_n, \\ \ell, & \tilde{g}(\ell) \ge \theta_n \text{ and } \tilde{g}(\ell + 1) < \theta_n. \end{cases} \tag{15}$$

3) Illustrative Examples: We use the example in Fig. 10 to illustrate the concept of the proposed algorithm. The utterance "call office" is first converted to log energy $g(t)$ and normalized to have the largest value be zero. For this example, the speech signal is about 2 s concatenated by another 2 s of heavy breath. We estimate the means and standard deviations of the speech and the background energy using the equations in Appendix B. The results are shown in Fig. 10, where lines A, B, C, and D indicate the estimated values of μ_v, θ_v, θ_n and μ_n, respectively. Then we computer $y_b(.)$ using (13). We note that the operation differs slightly from the real-time one. The result is shown in Fig. 11 as a solid line. After evaluating the values of the peaks that are above threshold θ_v, for this case, the locations of beginning points are first located at the centers of the highest peaks.

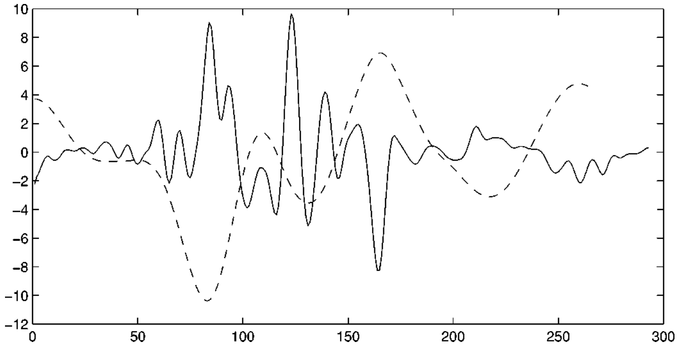

Fig. 11. Output of the beginning-edge filter (solid line) and ending-edge filter (dashed line).

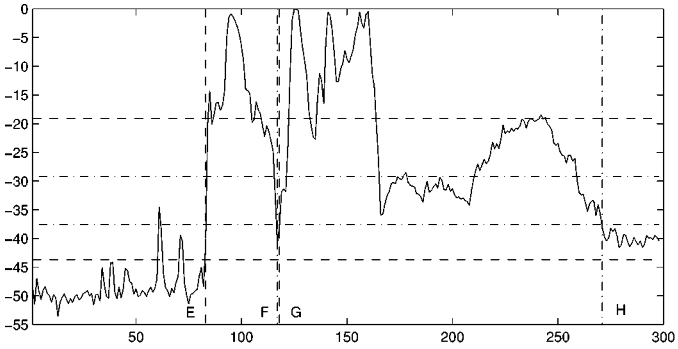

Fig. 12. Lines E, F, G, and H indicate the locations of two pairs of beginning and ending points.

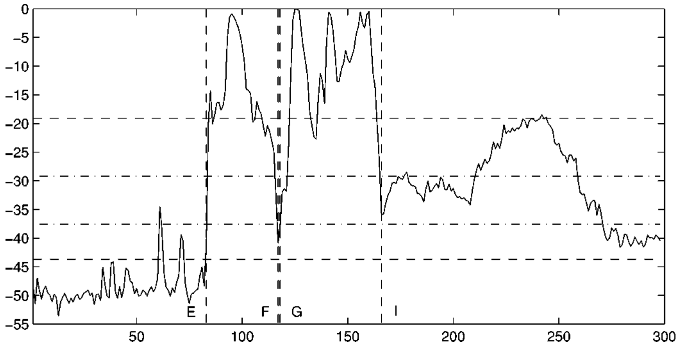

Fig. 13. Last ending point was adjusted from Line H to I by applying the ending-edge filter.

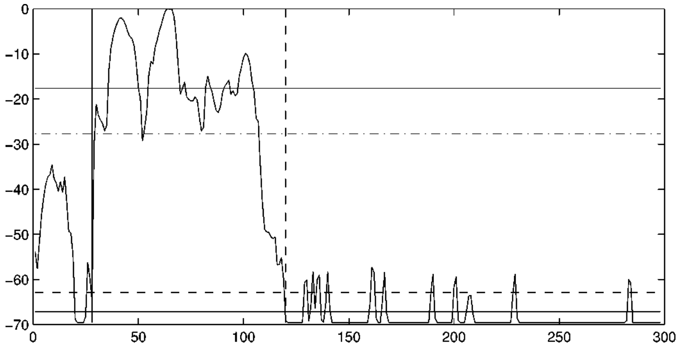

Fig. 14. Normalized log energy of "call Candice at her home" with breath in the beginning. The detected endpoints are the vertical solid line and dashed line.

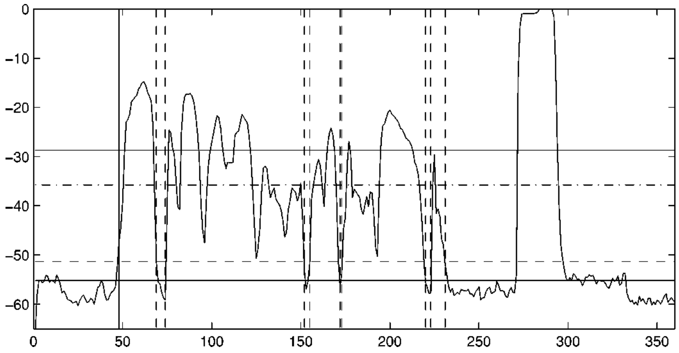

Fig. 15. Normalized log energy of "I pledge allegiance to the flag," with a dial tone at the end. The vertical lines indicate the beginning point and the ending points.

Actual beginning points can then be located by shifting the corresponding locations of the peaks to the left by half the filter size.

From the first beginning point, we search for the location of the corresponding ending point, where the energy level is lower than θ_n. For this example, we get two pairs of endpoints corresponding to two speech segments, as shown in Fig. 12, from line E to F and from line G to H, respectively. The clicks in the beginning of the utterances were not detected as speech because the filter responses at these locations were lower than the threshold value. As we can see from Fig. 12, the last segment between lines G and H includes the heavy breath.

The energy data in the segments are then fed into the ending-edge filter and compute (14). The filter output is shown in Fig. 11 as the dashed line. The ending point of the last segment is located by shifting the frame index of the largest peak to the right by about half the size of the ending edge filter. If the energy value is lower than θ_n at the shifted location, the ending point should be the last point where the energy level is greater than θ_n as described in (15). The final speech segments are from line E to line F and from line G to line I, as shown in Fig. 13.

More examples are shown in Figs. 14 and 15. Fig. 14 is the energy contour of utterance "Call Candice at her home phone" with breath in the beginning. The horizontal solid and dashed lines represent the means and thresholds for speech and noise. The detected beginning and ending points are shown as the vertical solid line and dashed lines, respectively. The breath signal is excluded from the speech segmentation successfully. Fig. 15 is an example of an utterance with a dial tone in the end. The first and the last vertical lines are the beginning and the ending points, respectively. Other vertical lines indicate the detected silence between words. The dial tone in the end of the utterance was detected and excluded from the speech segment.

B. Comparisons With HMM Forced-Alignment Approach

The database used for the comparison was collected for speaker verification with a common phrase "I pledge allegiance to the flag." It has 100 speakers and 4741 utterances in total. The utterances were collected over long distance telephone networks. The speakers were instructed to make the phone calls at different locations and using different telephone handsets. The collected utterances are with various kinds of noise. A pair of beginning and ending points was detected manually for every utterance. We use the manually detected endpoints

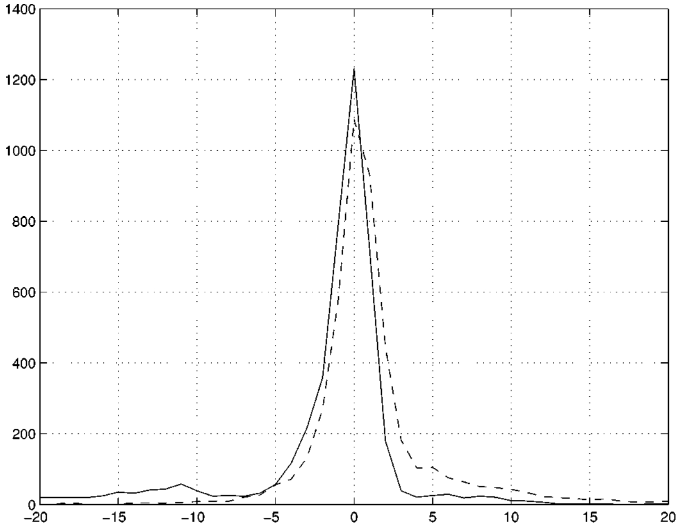

Fig. 16. Dashed line is the histogram of the differences between manually- and HMM-detected beginning points. Solid line is between manually and batch-mode detected beginning points.

as references to compare with the endpoints detected by the proposed approach and by HMM approach. Here, the HMM approach means endpoint detection by forced alignment assuming both the models and lexicons are available. The HMM approach uses 41 speaker-independent phoneme models. Each phone model has 3 states and each state has 32 Gaussian mixtures. The feature vector is composed of 12 cepstral and 12 delta cepstral coefficients. The cepstrum is derived from a tenth-order LPC analysis over a 30 ms window and the feature vectors are updated at 10 ms intervals.

The histogram of the differences between the manually detected beginning points and HMM detected beginning points is shown in Fig. 16 as a dashed line. The histogram of the differences between the manually-detected beginning points and the beginning points detected by the proposed approach is shown in Fig. 16 as a solid line. The statistics are listed in Table II. The accuracy of the proposed approach is very close to the HMM approach. The shift between those two histograms can be resolved by adjusting the thresholds in determining beginning points; however, it is not necessary since the overall difference between the two approaches is about the same.

The histogram of the differences between the manually detected ending points and HMM detected ending points is shown in Fig. 17 as a dashed line. The histogram of the differences between manually detected ending points and the ending points detected by the proposed approach is shown in Fig. 17 as a solid line. These two histograms are very close. We note that both of the histograms shift from the manually detected ending-points. This is due to the different interpretations on ending-points between human and algorithms.

TABLE II
STATISTICS OF THE DIFFERENCES ON DETECTED BEGINNING POINTS

Differences in no. of frames	0	≤ 1	≤ 2	≤ 3
Human vs. HMM	22.95%	54.82%	69.90%	76.52%
Human vs. Proposed	25.97%	57.84%	69.21%	74.58%

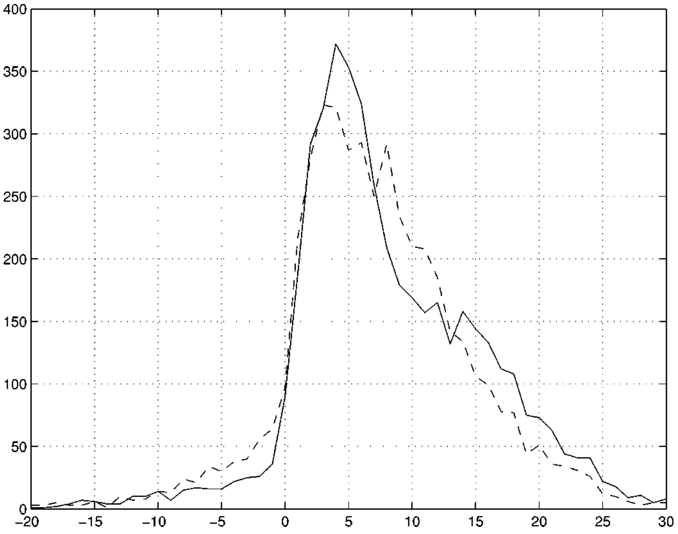

Fig. 17. Dashed line is the histogram of the differences between manually- and HMM-detected ending points. Solid line is between manually- and batch-mode detected ending points.

The histograms and table indicate that the endpoints detected by the proposed algorithm have the same accuracy as the HMM detected endpoints. Comparing with HMM approach, the proposed one does not need any language-dependent models and lexicon information; therefore, it can support language-independent applications. Also, the proposed algorithm is much faster. It only needs about 130 Kflops (floating point operations) for endpoint detection, while the HMM approach needs over 200 Mflops for forced alignment using a set of speaker-independent phoneme models. Furthermore, the proposed algorithm can detect the silence between words easily while it needs to involve much more computation when using the HMM approach.

C. Application to Language-Independent Speaker Verification

Since the proposed algorithm can detect endpoints at accuracy similar to the HMM approach, we apply the proposed algorithm to the front-end of a speaker verification system [2]. After LPC cepstral extraction, the proposed algorithm detects endpoints on the energy. Silence, breath, dial tone and other nonspeech signals are then removed from the feature set. Given the original feature observation of $\mathcal{O}$, after silence removal, the feature set becomes $\mathbf{O}$ which is a subset of $\mathcal{O}$, i.e., $\mathbf{O} \subset \mathcal{O}$. Cepstral mean subtraction (CMS) is then performed on $\mathbf{O}$.

This approach was evaluated on a database consisting of 38 speakers—18 male and 20 female for speaker verification (see [26] for the database descriptions). The common pass-phrase for all speakers is "call Janice at her office phone." Each true speaker was tested with the same pass-phrase from all impostors. In the language-independent configuration, the equal error rates (EERs) are 3.6% and 4.4% for male and female groups, respectively. In the language-dependent configuration where the background model is applied, the EERs are 2% and 3.5% for male and female groups, respectively. The average individual EER is 2.8%. The accuracy is in the same level as the speaker verification system where HMMs were applied to endpoint detection [27], [28].

The proposed algorithm has also been implemented in a real speech controller with embedded speaker verification. Readers are referred to [3] for detail.

V. Conclusions

In this paper, we propose two algorithms for real-time and batch-mode endpoint detection. Both algorithms apply filters to detect possible endpoints and then make final decisions based on the filter outputs. Since the filter is designed to be invariant to various levels of background noise, the proposed algorithms are reliable and robust, even in very low SNR situations.

In the real-time algorithm, a filter with a 24-frame look-ahead detects all possible endpoints. A three-state transition diagram then evaluates the output from the filter for final decisions. The detected endpoints are then applied to real-time energy normalization. Since the entire algorithm only uses a 1-D energy feature, it has low complexity and is very fast in computation. The evaluation in a noisy database has showed significant string error reduction, over 50% on all 5- to 20-dB SNR situations. The evaluations in telephone databases have showed over 30% reductions in four out of 12 databases. The proposed algorithm has been implemented in real-time ASR systems. The contributions are not only to improve the recognition accuracy but also the robustness of entire system in low SNR environments.

In the batch-mode algorithm, the peaks of the filter output are used to detect endpoints with thresholds estimated from a two-mixture energy distribution model, where the model parameters can be solved through closed-form equations. Using manually detected endpoints as references, we have compared the proposed algorithm with the forced-alignment approach using HMM. The experiments showed that the proposed algorithm has similar accuracy to the HMM approach while it needs much less computations. The algorithm has also been implemented in a real recognition system for language-independent speech control including embedded speaker verification [3].

Appendix A
Objective Function for the Optimal Filter Design

Assume that the beginning or ending edge in log energy is a ramp edge as defined in (2). And, assume that the edges are emerged with white Gaussian noise. Following Canny's criteria, Petrou and Kittler [13] derived the SNR for this filter $f(x)$ as being proportional to

$$S = \frac{\int_{-w}^{0} f(x)(1-e^{sx})\mathrm{d}x}{\sqrt{\int_{-w}^{0} |f(x)|^2 \mathrm{d}x}} \tag{16}$$

where w is a half width of the actual filter. They consider a good locality measure to be inversely proportional to the standard deviation of the distribution of endpoint where the edge is supposed to be. It was defined as

$$L = \frac{s^2 \int_{-w}^{0} f(x)e^{sx}\mathrm{d}x}{\sqrt{\int_{-w}^{0} |f'(x)|^2 \mathrm{d}x}}. \tag{17}$$

Finally, the measure for the suppression of false edges is proportional to the mean distance between the neighboring maxima of the response of the filter to white Gaussian noise

$$C = \frac{1}{w}\sqrt{\frac{\int_{-w}^{0} |f'(x)|^2 \mathrm{d}x}{\int_{-w}^{0} |f''(x)|^2 \mathrm{d}x}}. \tag{18}$$

Therefore, the combined objective function of the filter is

$$\begin{aligned} J =& \max_{f(x)} \left\{ (S \cdot L \cdot C)^2 \right\} \\ =& \frac{s^4}{w^2} \frac{\left| \int_{-w}^{0} f(x)(1-e^{sx})\mathrm{d}x \int_{-w}^{0} f(x)e^{sx}\mathrm{d}x \right|^2}{\int_{-w}^{0} |f(x)|^2 \mathrm{d}x \int_{-w}^{0} |f''(x)|^2 \mathrm{d}x}. \end{aligned} \tag{19}$$

Appendix B
Energy Model Estimation

Instead of the popular EM algorithm, we applied the moment algorithm [29] for a faster parameter estimation for the model in (11). Let $g = \{g_1, g_2, \ldots, g_n\}$ represent sample values. By equating the observed moments given by

$$V_r = \frac{1}{n}\sum_{i=1}^{n}(g_i - \bar{g})^r \qquad r = 0, 1, \ldots, 5 \tag{20}$$

where $\bar{g}$ is the sample mean to the theoretical moments given by

$$v_r = \int (g-\mu)^r f(g)\mathrm{d}g \qquad r = 0, 1, \ldots, 5 \tag{21}$$

where $\mu = E(g)$, we can obtain five nonlinear simultaneous equations and the solution has been summarized in [29]. To estimate the five parameters, we first find the real negative root of

$$\sum_{i=0}^{9} a_i u^i = 0 \tag{22}$$

where

$$\begin{aligned} a_9 =& 24, \qquad a_8 = 0 \\ a_7 =& 84k_4, \qquad a_6 = 36V_3^2 \\ a_5 =& 90k_4^2 + 72V_3k_5 \\ a_4 =& 444V_3^2k_4 - 18k_5^2 \\ a_3 =& 288V_3^4 - 108V_3k_4k_5 + 27k_4^3 \\ a_2 =& -(63V_3^2k_4^2 + 72V_3^3k_5) \\ a_1 =& -96V_3^4k_4 \\ a_0 =& -24V_3^6 \end{aligned}$$

and where $k_4 = V_4 - 3V_2^2$ and $k_5 = V_5 - 10V_3V_2$ are the fourth and fifth sample cumulates, respectively.

Let $\hat{u}$ be a real negative root of (22), then parameters $\hat{\delta_1}$ and $\hat{\delta_2}$ are obtained as roots of

$$\delta^2 - \frac{\hat{w}}{\hat{u}}\delta + \hat{u} = 0 \tag{23}$$

where

$$\hat{w} = \frac{-8V_3\hat{u}^3 + 3k_5\hat{u}^2 + 6V_3k_4\hat{u} + 2V_3^3}{2\hat{u}^3 + 3k_4\hat{u} + 4V_3^2}. \tag{24}$$

Now, the estimates of the five model parameters may be derived as the following explicit forms:

$$\hat{\mu}_1 = \bar{g} + \hat{\delta}_1 \tag{25}$$

$$\hat{\mu}_2 = \bar{g} + \hat{\delta}_2 \tag{26}$$

$$\hat{\sigma}_1^2 = \frac{1}{3}\hat{\delta}_1\left(\frac{2\hat{w}}{\hat{u}} - \frac{V_3}{\hat{u}}\right) + V_2 - \hat{\delta}_1^2 \tag{27}$$

$$\hat{\sigma}_2^2 = \frac{1}{3}\hat{\delta}_2\left(\frac{2\hat{w}}{\hat{u}} - \frac{V_3}{\hat{u}}\right) + V_2 - \hat{\delta}_2^2 \tag{28}$$

$$\hat{c} = \frac{\hat{\delta}_2}{\hat{\delta}_2 - \hat{\delta}_1}. \tag{29}$$

We note that $\mu_1 \neq \mu_2$ in the application. In the case that the solution of (22) does not exist, a histogram can be constructed to estimate the mixture model parameters approximately.

Acknowledgment

The authors wish to thank Dr. D. W. Tufts for a very helpful lecture and Drs. W.-G. Kim, C.-H. Lee, O. Siohan, F. K. Soong, and F. Korkmazskiy for useful discussions.

References

[1] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition* . Englewood Cliffs, NJ: Prentice-Hall, 1993.

[2] Q. Li and A. Tsai, "A matched filter approach to endpoint detection for robust speaker verification," in *Proc. IEEE Workshop on Automatic Identification*, Summit, NJ, Oct. 1999.

[3] ——, "A language-independent personal voice controller with embedded speaker verification," in *Proc. Eurospeech'99*, Budapest, Hungary, Sept. 1999.

[4] K. Bullington and J. M. Fraser, "Engineering aspects of TASI," *Bell Syst. Tech. J.*, pp. 353–364, Mar. 1959.

[5] J. G. Wilpon, L. R. Rabiner, and T. Martin, "An improved word-detection algorithm for telephone-quality speech incorporating both syntactic and semantic constraints," *AT&T Bell Labs. Tech. J.*, vol. 63, pp. 479–498, Mar. 1984.

[6] R. Chengalvarayan, "Robust energy normalization using speech/nonspeech discriminator for German connected digit recognition," in *Proc. Eurospeech'99*, Budapest, Hungary, Sept. 1999, pp. 61–64.

[7] J. A. Haigh and J. S. Mason, "Robust voice activity detection using cepstral features," in *Proc. IEEE TENCON*, 1993, pp. 321–324.

[8] L. R. Rabiner and M. R. Sambur, "An algorithm for determining the endpoints of isolated utterances," *Bell Syst. Tech. J.*, vol. 54, pp. 297–315, Feb. 1975.

[9] J. C. Junqua, B. Reaves, and B. Mak, "A study of endpoint detection algorithms in adverse conditions: Incidence on a DTW and HMM recognize," in *Proc. Eurospeech*, 1991, pp. 1371–1374.

[10] L. F. Lamel, L. R. Rabiner, A. E. Rosenberg, and J. G. Wilpon, "An improved endpoint detector for isolated word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 777–785, Aug. 1981.

[11] S. G. Tanyer and H. Özer, "Voice activity detection in nonstationary noise," *IEEE Trans. Speech Audio Processing*, vol. 8, pp. 478–482, July 2000.

[12] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 679–698, Nov. 1986.

[13] M. Petrou and J. Kittler, "Optimal edge detectors for ramp edges," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 483–491, May 1991.

[14] E. Carlstein, M. Muller, and D. Siegmund, *Change-Point Problems* . Hayward, CA: Inst. Math. Statist., 1994.

[15] R. K. Bansal and P. Papantoni-Kazakos, "An algorithm for detecting a change in stochastic process," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 227–235, Mar. 1986.

[16] A. Wald, *Sequential Analysis* . London, U.K: Chapman & Hall, 1947.

[17] Q. Li, "A detection approach to search-space reduction for HMM state alignment in speaker verification," *IEEE Trans. Speech Audio Processing*, vol. 9, pp. 569–578, July 2001.

[18] B. Brodsky and B. S. Darkhovsky, *Nonparametric Methods in Change-Point Problems* . Norwell, MA: Kluwer, 1993.

[19] B. S. Atal, "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification," *J. Acoust. Soc. Amer.*, vol. 55, pp. 1304–1312, 1974.

[20] ——, "Automatic recognition of speakers from their voices," *Proc. IEEE*, vol. 64, pp. 460–475, 1976.

[21] S. Furui, "Cepstral analysis techniques for automatic speaker verification," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 254–277, Apr. 1981.

[22] L. A. Spacek, "Edge detection and motion detection," *Image Vision Comput.*, vol. 4, pp. 43–43, 1986.

[23] Q. Li, J. Zheng, Q. Zhou, and C.-H. Lee, "A robust, real-time endpoint detector with energy normalization for ASR in adverse environments," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Salt Lake City, UT, May 2001.

[24] W. Chou, C.-H. Lee, and B.-H. Juang, "Minimum error rate training of inter-word context dependent acoustic model units in speech recognition," in *Proc. Int. Conf. on Spoken Language Processing*, 1994, pp. 432–439.

[25] C.-H. Lee, E. Giachin, L. R. Rabiner, R. Pieraccini, and A. E. Rosenberg, "Improved acoustic modeling for large vocabulary speech recognition," *Comput. Speech Lang.*, vol. 6, pp. 103–127, 1992.

[26] R. A. Sukkar and C.-H. Lee, "Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 4, pp. 420–429, Nov. 1996.

[27] S. Parthasarathy and A. E. Rosenberg, "General phrase speaker verification using sub-word background models and likelihood-ratio scoring," in *Proc. ICSLP-96*, Philadelphia, PA, Oct. 1996.

[28] Q. Li, S. Parthasarathy, and A. E. Rosenberg, "A fast algorithm for stochastic matching with application to robust speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Munich, Germany, Apr. 1997, pp. 1543–1547.

[29] B. S. Everitt and D. J. Hand, *Finite Mixture Distributions*. London, U.K.: Chapman & Hall, 1981.

Qi (Peter) Li (S'87–M'88–SM'01) received the Ph.D. degree in electrical engineering from the University of Rhode Island, Kingston, in 1995.

In 1995, he joined Bell Laboratories, Murray Hill, NJ, where he is currently a Member of Technical Staff in the Dialogue Systems Research Department. From 1988 to 1994, he was with F.M. Engineering and Research, Norwood, MA, where he worked in research on patent recognition algorithms and in real-time systems. His research interests include robust speaker and speech recognition, robust feature extraction, fast search algorithms, stochastic modeling, fast discriminative learning, and neural networks. His research results have been implemented in Lucent products. He has published extensively, filed and awarded many patents in his research areas.

Dr. Li has been active as a reviewer for several journals, including IEEE Transactions on Speech and Audio Processing, as a Local Chair for the IEEE 1999 Workshop on Automatic Identification, and as a committee member for several IEEE workshops. He has received two awards and is listed in *Who's Who in America* (Millennium and 2001 Editions).

Jinsong Zheng received the B.S. and M.S. degrees in computer science from Fudan University, Shanghai, China, and Utah State University, Logan, UT, respectively.

Between 1994 and 1998, he was a Software Engineer with WebSci Technologies, South Brunswick, NJ. Since 1998, he has been a Consultant in the Dialogue Systems Research Department of Bell Labs, Lucent Technologies, Murray Hill, NJ, where he has been involved in various research projects in speech recognition. He is also a member of the Lucent Automatic Speech Recognition (LASR) software development team.

Augustine Tsai received the M.S. degree in systems engineering from Case Western Reserve University, Cleveland, OH, in 1989, a second M.S. degree and the Ph.D degree in electrical engineering from Rutgers University, New Brunswick, NJ, in 1991 and 1996, respectively.

He was a Lead Engineer with the U.S. Army Face Recognition Project while he was with CAIP, Rutgers University, in 1994–1995. He was with SpeakEZ developing speaker verification products. In 1997, he worked on the ATM LAN Emulation for MPEG II video broadcast in AT&T Since 1998, he has been with the Multimedia Communication Research Laboratory, Bell Labs, Murray Hill, NJ. He has been involved with various activities in speaker verification, dialogue systems, and language modeling. He has contributed in design of the dialogue session manager for the VoiceXML platform. He is currently working on QoS policy management in the multiprotocol label switching (MPLS) based media networks. He has publications in machine vision, face recognition, speech/3-D audio processing, and video networks.

Qiru Zhou (S'86–M'92) received the B.S. and M.S. degrees in electrical and computer engineering from Northern Jiao-Tong University, China, and Beijing University of Posts and Telecommunications, China, respectively.

He joined Bell Labs, AT&T, in 1992. Currently he is a Member of Technical Staff at Bell Labs, Lucent Technologies, Murray Hill, NJ, with the Dialogue Systems Research Department. His research interests include speech and speaker recognition algorithms, multimodal dialogue infrastructure, real-time distributed object-oriented software methodology for multimedia communication systems, and standard for speech and multimedia applications. Since 1992, He has been involved and lead in various projects in AT&T and Lucent to apply speech and dialogue technologies into products. He is now a Technical Leader in Lucent speech software product development.