LOW-POWER SVM CLASSIFIERS FOR SOUND EVENT CLASSIFICATION ON MOBILE DEVICES

Man-Wai Mak

Dept. of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hong Kong

Sun-Yuan Kung

Dept. of Electrical Engineering
Princeton University, USA

ABSTRACT

With the high processing power of today's smartphones, it becomes possible to turn a smartphone into a personal audio surveillance and monitoring system. Ideally, such a system should be able to detect and classify a variety of sound events 24 hours a day and trigger an emergence phone call or message once a specified sound event (e.g., screaming) occurs. To prolong battery life, it is important to trade off the detection accuracy against power consumption. This paper investigates the power consumption of different stages of a sound-event classification system, including segmentation, feature extraction, and SVM scoring. The performance and power consumption of various acoustic features and SVM kernels are compared. This paper advocates the notion of *intrinsic complexity* through which the scoring function of polynomial SVMs can be written in a matrix-vector-multiplication form so that the resulting complexity becomes independent of the number of support vectors. Results show that this intrinsic complexity can reduce the CPU utilization of polynomial SVMs by 28 times without reducing classification accuracy.

Index Terms— Low-power SVM; kernel-energy tradeoff; sound event classification; smartphones; audio surveillance.

1. INTRODUCTION

Surveillance and monitoring for security purposes have become increasingly important in today's society. Traditionally, surveillance is achieved by recording the scenes via video cameras. However, taking video is inappropriate (e.g., in a washroom) or even impossible (e.g., insufficient lighting) under some situations. Audio is a viable alternative under such situations. In fact, audio is more effective than video in detecting the events that posses unique acoustic signatures [1–3], e.g., screaming, crying, gunshots, door slam, etc. Moreover, video-based surveillance typically requires using fixed cameras to record the scenes, restricting the surveillance system to be localized in one place. However, audio-based surveillance can make effective use of mobile devices, allowing the surveillance system to be moved from one place to another easily.

Sound event detection (SED) [4, 5] is a branch of computational auditory scene analysis [6] where computational methods are used to separate and recognize mixtures of sounds in natural environments similar to what human listeners can do. Recently, sound event detection has attracted more attention from the research community, primarily because of the increasing importance in security. SED has also attracted attention in the speech recognition community where the goal is to recognize not only speech but also other sound events such as paper flipping, breathing, and coughing during a meeting [7]. The models and features (e.g. HMM and MFCC) being used, however, are mainly borrowed from speech recognition, as the main goal is still to recognize the speech in a meeting.

The recent ubiquity of smartphones has opened up new applications of sound event detection. For example, any smartphones can be turned into a personal security device, allowing users to detect any hazardous situations around them 24 hours a day. Abnormal sound events such as screaming can be detected and emergency phone calls can be automatically made. This kind of personal security device is particularly useful for children and women who are more vulnerable to assaults. The ability to detect scream sounds and body fall is also useful for the elderly.

However, audio monitoring using mobile devices has a catch. Because monitoring should be carried out continuously, the monitoring process will easily drain the battery of the smartphone if the detection algorithm is not carefully designed to reduce power consumption. To minimize power consumption, spectral analysis should be avoided when there is no sound events (i.e., only background signals present). Therefore, a low-power sound activity detector that does not require spectral analysis is desirable. To this end, the average energy and zero-crossing rate are ideal choices. When a potential sound segment is found, it is necessary to classify the type of sound events for further action. For example, if it is a scream, a phone call will be automatically made. To classify sound events, energy level and zero-crossing rate will not be sufficient, and spectral analysis is required. As will be demonstrated in this paper, spectral analysis consumes significantly more battery power than time-domain analysis. It is therefore necessary to investigate the power consumption of various spectral-based techniques.

Another source of power consumption is the classification process. The popular Gaussian mixture model or hidden Markov model are not appropriate because of the complexity in the likelihood function. Linear SVMs are ideal but their capability is limited. Nonlinear SVMs strike a good compromise between power consumption and classification ability. Here, we propose a low-power polynomial SVM that takes advantage of the fact that part of the SVM scoring process can be pre-computed during off-line training. We derive this new formulation and refer to its complexity as *intrinsic complexity* of SVMs. We also make use of the symmetric property of the multinomial coefficients in the polynomial expansion to reduce the intrinsic complexity further.

The paper is organized as follows. Section 2 explains how the intrinsic complexity of polynomial SVMs can help reduce computation during the scoring stage. Then, Sections 3 and 4 demonstrate the merit of intrinsic complexity via performance analysis and CPU utilization analysis of a sound-event classification task.

This work was in part supported by The Hong Kong Research Grant Council, Grant No. PolyU5251/08E and PolyU5264/09E.

2. LOW-POWER SUPPORT VECTOR MACHINES

2.1. Intrinsic Complexity of Polynomial SVM

Assume that we are given N training vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of dimension M with labels $y_n \in \{+1, -1\}$, $n = 1, \ldots, N$. Using the pairs $\{\mathbf{x}_n, y_n\}_{n=1}^N$, a support vector machine (SVM) can be trained. Given a test vector $\mathbf{x}$, the SVM's output is written as

$$\begin{aligned} f(\mathbf{x}) &= \sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}, \mathbf{x}_n) + b \\ &= \sum_{n=1}^{N} a_n K(\mathbf{x}, \mathbf{x}_n) + b, \end{aligned} \tag{1}$$

where $a_n \equiv \alpha_n y_n$, b is a bias term, α_n's are Lagrange multipliers, and $K(\mathbf{x}, \mathbf{x}_n)$ is a kernel function.

When $K(\mathbf{x}, \mathbf{x}_n)$ is a 2nd-order polynomial kernel, we may express Eq. 1 in a matrix-vector-multiplication form [8]:

$$f(\mathbf{x}) = \sum_{n=1}^{N} a_n \left(1 + \frac{\mathbf{x} \cdot \mathbf{x}_n}{\sigma^2}\right)^2 + b \tag{2a}$$

$$= \sum_{n=1}^{N} a_n \left(\tilde{\mathbf{x}}^\mathsf{T} \tilde{\mathbf{x}}_n\right)^2 + b \tag{2b}$$

$$= \tilde{\mathbf{x}}^\mathsf{T} \left(\sum_{n=1}^{N} a_n \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\mathsf{T}\right) \tilde{\mathbf{x}} + b \tag{2c}$$

$$= \tilde{\mathbf{x}}^\mathsf{T} \tilde{\mathbf{W}} \tilde{\mathbf{x}} + b, \tag{2d}$$

where $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{W}}$ are $(M+1)$-dimensional vector and $(M+1) \times (M+1)$ symmetric matrix defined respectively as

$$\tilde{\mathbf{x}} \equiv \begin{bmatrix} \mathbf{x}\sigma^{-1} \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{W}} \equiv \sum_{n=1}^{N} a_n \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\mathsf{T},$$

where σ is a scaling constant. Note that because $\tilde{\mathbf{W}}$ depends only on a_n's and training data $\mathbf{x}_n$, it can be pre-computed during training. The decision boundary is determined by this $(M+1) \times (M+1)$ matrix, whose size is independent of the training size N.

To study the complexity of Eq. 2, let us further rewrite it as

$$f(\mathbf{x}) = \sum_{n=1}^{N} a_n \left(\tilde{\mathbf{x}}^\mathsf{T} \tilde{\mathbf{x}}_n\right)^2 + b \tag{3a}$$

$$= \sum_{n=1}^{N} a_n \left(\sum_{i=1}^{M+1} \tilde{x}^{(i)} \tilde{x}_n^{(i)}\right) \left(\sum_{j=1}^{M+1} \tilde{x}^{(j)} \tilde{x}_n^{(j)}\right) + b \tag{3b}$$

$$= \sum_{i=1}^{M+1} \sum_{j=1}^{M+1} \tilde{w}_{ij} \tilde{x}^{(i)} \tilde{x}^{(j)} + b \tag{3c}$$

$$= \sum_{i=1}^{M+1} \tilde{x}^{(i)} \left(\sum_{j=1}^{M+1} \tilde{w}_{ij} \tilde{x}^{(j)}\right) + b, \tag{3d}$$

where $\tilde{w}_{ij} \equiv \sum_{n=1}^{N} a_n \tilde{x}_n^{(i)} \tilde{x}_n^{(j)}$ and $\tilde{x}^{(i)}$ is the i-th element of $\tilde{\mathbf{x}}$. The summation within the inner parentheses in Eq. 3d amounts to $(M+1)^2$ operations which account for the majority of the operations. Note that the number of operations for the subsequent summation is $(M+1)$ and thus can conveniently be neglected in our estimate. Furthermore, by exploiting the symmetry of $\tilde{\mathbf{W}}$, Eq. 3d can be further reduced to

$$f(\mathbf{x}) = \sum_{i=1}^{M+1} \tilde{x}^{(i)} \left(\sum_{j=1}^{i} m_{ij} \tilde{w}_{ij} \tilde{x}^{(j)}\right) + b \tag{4}$$

where m_{ij} are the binomial coefficients. More precisely, $m_{ij} = \frac{2!}{\gamma_1! \cdots \gamma_{M+1}!}$, where γ_l $(l = 1, \ldots, M+1)$ is the number of occurrences of l in $\{i, j\}$ and $\sum_{l=1}^{M+1} \gamma_l = 2$. This means that the number of operations can be further reduced to

$$J_2 = \frac{(M+1)(M+2)}{2!} = \binom{M+2}{2} \approx \frac{(M+1)^2}{2}. \tag{5}$$

More importantly, the computational complexity has now become independent of N (or the number of support vectors S in the case of SVM) during the on-line classification/detection phase.

Let's extend the above concept to p-th order polynomial SVMs:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{n=1}^{N} a_n \left(\tilde{\mathbf{x}}^\mathsf{T} \tilde{\mathbf{x}}_n\right)^p + b \\ &= \sum_{n=1}^{N} a_n \left(\sum_{i_1=1}^{M+1} \tilde{x}^{(i_1)} \tilde{x}_n^{(i_1)}\right) \cdots \left(\sum_{i_p=1}^{M+1} \tilde{x}^{(i_p)} \tilde{x}_n^{(i_p)}\right) + b \\ &= \sum_{i_1=1}^{M+1} \sum_{i_2=1}^{M+1} \cdots \sum_{i_p=1}^{M+1} \tilde{w}_{i_1 \cdots i_p} \tilde{x}^{(i_1)} \tilde{x}^{(i_2)} \cdots \tilde{x}^{(i_p)} + b \\ &= \sum_{i_1=1}^{M+1} \tilde{x}^{(i_1)} \left[\sum_{i_2=1}^{M+1} \tilde{x}^{(i_2)} \left(\cdots \sum_{i_p=1}^{M+1} \tilde{w}_{i_1 \cdots i_p} \tilde{x}^{(i_p)}\right)\right] + b \\ &= \sum_{i_1=1}^{M+1} \tilde{x}^{(i_1)} \left[\sum_{i_2=1}^{i_1} \tilde{x}^{(i_2)} \left(\cdots \sum_{i_p=1}^{i_{p-1}} m_{i_1 \cdots i_p} \tilde{w}_{i_1 \ldots i_p} \tilde{x}^{(i_p)}\right)\right] + b \end{aligned}$$

where $\tilde{w}_{i_1 \cdots i_p} \equiv \sum_{n=1}^{N} a_n \tilde{x}_n^{(i_1)} \cdots \tilde{x}_n^{(i_p)}$ and $m_{i_1 \cdots i_p} = \frac{p!}{\gamma_1! \cdots \gamma_{M+1}!}$ are multinomial coefficients. Generalizing from the 2nd-order case, $\gamma_l, l = 1, \ldots, M+1$, is the number of occurrences of l in $\{i_1, \ldots, i_p\}$ and $\sum_{l=1}^{M+1} \gamma_l = p$. Therefore, for polynomial kernel of degree p, the number of operations is

$$J_p = \binom{M+p}{p} \approx \frac{(M+1)^p}{p!}. \tag{6}$$

In summary, we have introduced two different costs for the computation of the decision function: (1) according to its *intrinsic* complexity $\mathcal{O}\left((M+1)^p/p!\right)$, which is independent of the number of training samples N; and (2) by means of direct kernel evaluation through Eq. 2a, which has a complexity of $\mathcal{O}(MN)$. A cost conscientious choice will adopt the less of the two options, leading to

$$\text{Computational-Complexity} = \min\left\{\eta MN, \binom{M+p}{p}\right\},$$

where $0 \leq \eta \leq 1$ indicates the fraction of nonzero coefficients in $\{a_n,\ n = 1, \ldots, N\}$. In case of SVMs, $\eta = \frac{S}{N} < 1$, as only the support vectors have nonzero α_n. Thus, the minimum computational complexity becomes

$$\text{Computational-Complexity} = \min\left\{MS, \binom{M+p}{p}\right\}. \tag{7}$$

Compared with Other Methods. To reduce the run-time com-

plexity of SVMs, Burges proposed the reduced-set method [9] that approximates Eq. 1 by replacing the full set of support vectors with a reduced set of non-support vectors. Our method is different in that instead of approximating Eq. 1, our approach produces the same decision function as the one that uses the full set of support vectors but with significant reduction in run-time complexity.

2.2. Complexity of RBF-SVM

Unlike polynomial kernels, RBF kernels cannot be expressed in a compact form similar to Eq. 2(d). In fact, for RBF kernels, the number of operations to compute the inner products in the kernel-induced feature space is infinite, i.e., $J \rightarrow \infty$. Of course, we can use the kernel trick to compute the inner product without accessing the infinite dimensional space. But this still amounts to $\mathcal{O}(MS)$ operations, where M is the dimension of $\mathbf{x}$ and S is the number of support vectors. This means that the complexity of RBF-SVM grows linearly with the number of support vectors. This could become a computational hurdle when $S \gg M$. In comparison, the complexity of low-order polynomial kernel (e.g., Eq. 5) is determined by its intrinsic complexity which does not scale with S. This could result in an enormous computation (energy) saving and thus become an influencing factor in the choice of kernels in tradeoff consideration.

3. EXPERIMENTS

Data Collection. We collected 103 sound events and an audio recording of a train station. The sound events are divided into six categories shown in Table 1. All events were sampled at 8kHz, 16 bits per sample. The sound events were added to the train-station recording at an SNR of -5dB, 0dB, 5dB, and 10dB.

Sound Event Segmentation. The sound-event classification is divided into three stages: segmentation, feature extraction, and SVM scoring. In real-time sound event classification, segmentation will be performed continuously, whereas feature extraction and scoring will be performed once a sound segment is found. To implement such a system on mobile devices, it is imperative to use the least power-consuming segmentation method. To this end, we used average energy for detecting potential sound events. Specifically, at any frame, the average energy of the current and the previous 3 frames was calculated and compared with a decision threshold, which was estimated from non-sound event periods. The segmentation algorithm tracks the moving average continuously to detect any potential sound segments. The acoustic features of the event segments are then extracted. Fig. 1 shows an example of event segmentation (3rd panel) and feature vectors (4th and 5th panel) of the detected sound segments.

Acoustic Feature Extraction. We have investigated four acoustic features, including MPEG-7 audio spectral flatness (ASF) [10], mel-frequency cepstral coefficients (MFCC) [11], linear-prediction cepstral coefficients (LPCC), and Mel filter-bank spectrum (Mel-spec). The frame size and frame rate were set to 32ms and 31.25Hz, respectively. For ASF, MFCC and Mel-spectrum, a 256-point FFT was applied to each frame. The dimension of ASF vectors is 15. For all other features, the dimension is 12, i.e., $M = 12$. To avoid extra computation burden on the SVM scoring phase, we did not use delta features. All acoustic vectors were subjected to z-norm before applying to the SVM classifiers.

Classifier Setting. For the RBF-SVM, we set the RBF width to 1 and penalty factor C to 1 ($C = 1 \sim 5$ give almost the same results). For Poly-SVM, we set degree $p = 2$, $\sigma = \sqrt{2}$ in Eq. 2a, and penalty factor $C = 0.01$.

Performance Evaluation and Metrics. An important performance measure for mobile applications is power consumption. As it is difficult to measure the power on a mobile devices, we used the percentage of CPU utilization reported from the task manager of a smartphone (Acer neoTouch S200 equipped with an ARMv7-A processor) as the metric for power consumption.

Sound Event	No. of Samples	Length (sec.)
Scream	52	0.5 – 5.8
Door-slam	8	0.1 – 0.4
Gunshot	8	0.4 – 2.0
Baby cry	16	1.2 – 2.5
Speech	16	0.3 – 1.7
Phone-ring	9	1.2 – 9.0
Total	103	1942

Table 1. Summary of sound events used in the experiments.

We used leave-one-out cross validation to compare the performance of polynomial and RBF SVMs under different signal-to-noise ratios. Specification, for each fold, one sound sample (file) is left out as test data, and the remaining 102 samples were used for training. Two scenarios were considered: (1) scream detection and (2) sound event classification. For the former, the scream event was considered as the target-class whereas the other five sound events were considered as the nontarget-class. For the latter, for each fold, a one-vs-rest SVM classifier was trained to classify the test sound file into one of the six sound classes.

4. RESULTS AND DISCUSSIONS

CPU Utilization. Table 2 shows the CPU utilization of various processes in the sound-event classification task. Evidently, frequency-domain features such as LPCC and MFCC consume significantly more CPU time (power) than time-domain features such as energy and zero-crossing rate. However, time-domain feature extraction needs to be performed continuously whereas LPCC and MFCC only need to be extracted after a potential sound segment is detected.

Among the classifiers investigated, RBF-SVM is the most power demanding, primarily because of its high complexity ($\mathcal{O}(MS)$). The CPU utilization of fast Poly-SVM scoring is only $1/28$ of that of conventional SVM scoring. This low CPU utilization clearly demonstrates the significant computation saving achievable by pre-computing the weights of polynomial SVMs in Eq. 3d.

Table 2 shows that under the worse case scenario where sound events occur at all time, the CPU utilization is about 18% (16% for LPCC, 1% for energy estimation, and 1% for fast Poly-SVM scoring). However, this situation is rare and we may comfortably assume that sound events occur at 10% of the time under normal situation. Under this normal situation, the CPU utilization becomes 4% if conventional Poly-SVM is used. This is further reduced to 2% when fast Poly-SVM is used, which represents a two-fold in power reduction.

Table 2 also shows the CPU utilization of Windows Media Player. Evidently, under normal operating condition, our proposed system consumes the same amount of power as the media player. If sound events occur less often than 10% of the time, the detection system will consume even less energy than the media player.

Poly-SVM vs. RBF-SVM. Fig. 2 shows the accuracy of sound event classification and the equal error rate (EER) of scream detection using various acoustic features as input vectors to polynomial and RBF SVMs. Note that both ordinary and fast Poly-SVM give the same accuracy and EER. Evidently, in both detection and classification tasks, polynomial SVMs generally outperform RBF-SVMs. However, none of the features stands out across different SNR and

Process	CPU Utilization
12-dim LPCC	16%
12-dim MFCC	37%
12-dim Mel filter-bank spectrum	25%
15-dim MPEG-7 ASF	25%
Energy	$< 1\%$
Zero-crossing rate	$< 1\%$
RBF-SVM scoring ($\mathcal{O}(MS)$, $S = 1886$)	35%
Poly-SVM scoring ($\mathcal{O}(MS)$, $S = 2096$, Eq. 3a)	28%
Fast Poly-SVM scoring ($\mathcal{O}((M+1)^2/2)$, Eq. 4)	$< 1\%$
Scream Detection System with Poly-SVM	4%
Scream Detection System with fast Poly-SVM	2%
Windows Media Player (Audio only)	2%

Table 2. CPU (ARMv7-A in Acer neoTouch S200) utilization of various processes in the sound-event detection task. Except for ASF, $M = 12$ for all cases. S is the average number of support vectors in the SVMs using LPCC vectors extracted from sound events with 0dB SNR. The scream detection system assumes that LPCC are used as features and that sound events occur 10% of the time.

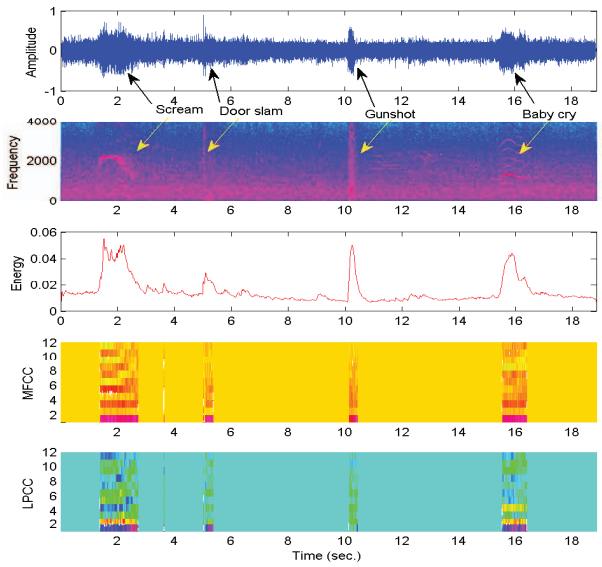

Fig. 1. Waveform, spectrogram, energy profile, MFCC, and LPCC of 4 sound events in train-station background at an SNR of 0dB.

types of SVMs. Therefore, if power consumption is a concern, we may select LPCC as it requires the least resources to compute and select fast Poly-SVMs due to its low intrinsic complexity (see Table 2).

5. REFERENCES

[1] C. Clavel, T. Ehrette, and G. Richard, "Events detection for an audio-based surveillance system," in *IEEE Int. Conf. on Multimedia and Expo*, 2005, pp. 1306–1309.

[2] L. Gerosa, G. Valenzise, F. Antonacci, M. Tagliasacchi, and A. Sarti, "Scream and gunshot detection in noisy environments," in *15th European Signal Processing Conference*, Poznan, Poland, Sep. 3-7 2007.

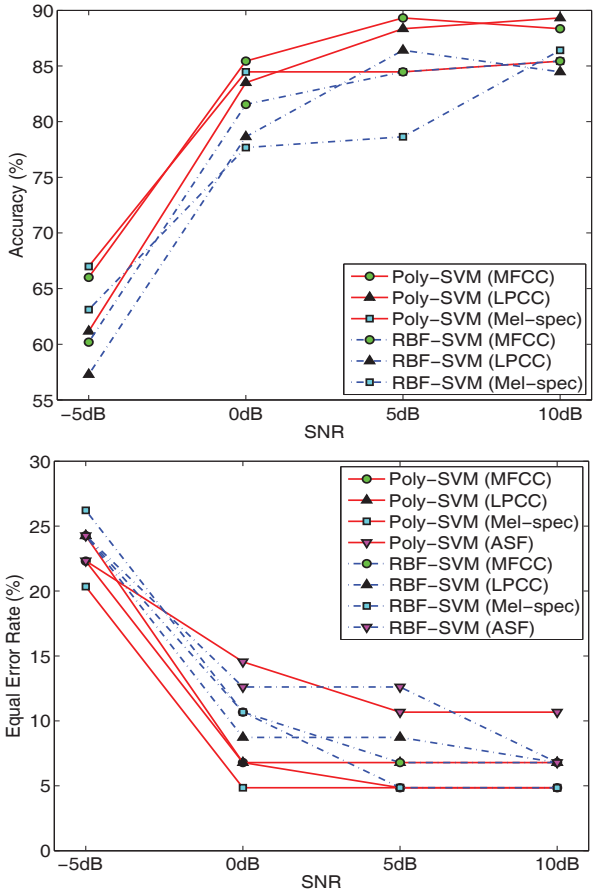

Fig. 2. *Top*: Accuracy against SNR for various acoustic features in the sound-event classification task. *Bottom*: Equal error rate (false alarm rate equal to miss probability) against SNR in the scream-detection task.

[3] S. Ntalampiras, I. Potamitis, and N. Fakotakis, "On acoustic surveillance of hazardous situations," in *ICASSP'2009*, 2009, pp. 165–168.

[4] P.K. Atrey, N.C. Maddage, and M.S. Kankanhalli, "Audio based event detection for multimedia surveillance," in *IEEE Int. Conference on Acoustics, Speech and Signal Processing*, 2006, vol. 5, pp. 813–816.

[5] H.D. Tran and H. Li, "Sound event recognition with probabilistic distance SVMs," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1556–1568, 2011.

[6] D.L. Wang and G.J. Brown, *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*, IEEE Press, 2006.

[7] R. Stiefelhagen and J.S. Garofolo, *Multimodal technologies for perception of humans: First International Evaluation Workshop on Classification of Events, Activities and Relationships, CLEAR 2006, Southampton, UK, April 6-7, 2006: revised selected papers*, vol. 4122, Springer-Verlag New York Inc, 2007.

[8] S.Y. Kung, *Kernel-Based Machine Learning*, Cambridge University Press, to appear.

[9] Chris J.C. Burges, "Simplified support vector decision rules," in *Proc. ICML'96*, 1996, pp. 71–77.

[10] H.G. Kim, N. Moreau, and T. Sikora, *MPEG-7 Audio and Beyond*, Wiley, 2005.

[11] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. on ASSP*, vol. 28, no. 4, pp. 357–366, 1980.