한국음향학회지 제30권 제2호 pp. 100~106 (2011)

Decorrelated Filter Bank를 이용한 음악 장르 분류 시스템

Music Genre Classification System Using Decorrelated Filter Bank

임 신 철*, 장 세 진**, 이 석 필**, 김 무 영*
(Shin-Cheol Lim*, Sei-Jin Jang**, Seok-Pil Lee**, Moo Young Kim*)

*세종대학교 정보통신공학과, **전자부품연구원 디지털미디어연구센터
(접수일자: 2011년 1월 30일; 수정일자: 2011년 2월 15일; 채택일자: 2011년 2월 16일)

음원의 디지털화가 진행되면서 음악 데이터베이스가 방대해지고 있다. 따라서, 음악 데이터를 보다 효과적으로 관리하기 위해 음악의 특성에 따라 장르별로 자동 분류해주는 시스템이 필요하다. 기존 장르 분류 시스템은 대부분 Mel-Frequency Cepstral Coefficient (MFCC)를 특징 벡터로 이용하고 있다. 본 논문에서는 Auditory Filter Bank를 이용한 Decorrelated Filter Bank (DFB)와 Octave-based Spectral Contrast (OSC)에 texture window를 적용하여 특징을 추출한 후, Support Vector Machine (SVM)을 이용하여 장르 분류를 시도하였다. 기존의 Marsyas 장르 분류 시스템과 비교한 결과 DFB와 OSC로 복합적인 특징 벡터를 구성하면 더 적은 차수의 특징벡터를 사용함에도 4.2 %의 향상된 분류 성공률을 얻을 수 있었다.
핵심용어: 장르 분류, Texture Window, SVM, OSC, DFB, MFCC
투고분야: 음악음향 및 음향심리 분야 (8.6)

Music recordings have been digitalized such that huge size of music database is available to the public. Thus, the automatic classification system of music genres is required to effectively manage the growing music database. Mel-Frequency Cepstral Coefficient (MFCC) is a popular feature vector for genre classification. In this paper, the combined super-vector with Decorrelated Filter Bank (DFB) and Octave-based Spectral Contrast (OSC) using texture windows is processed by Support Vector Machine (SVM) for genre classification. Even with the lower order of the feature vector, the proposed super-vector produces 4.2 % improved classification accuracy compared with the conventional Marsyas system.
Keywords: Genre Classification, Texture Window, SVM, OSC, DFB, MFCC
ASK subject classification: Musical Acoustics and Psychoacoustics (8.6)

I. 서 론

최근 수많은 음악이 제작되고 있는 가운데 음악 데이터를 어떻게 효과적으로 분류하는지가 이슈가 되고 있다. 기존에는 수작업으로 음악 데이터를 분류하였다. 하지만, 방대한 디지털 음원들을 분류하기 위해서는 작곡가, 가수, 장르 별로 자동 분류하는 알고리즘이 필요하다. 본 논문에서는 장르별 음원 자동 분류 알고리즘을 제안하고자 한다.

책임저자: 김 무 영 (mooyoung@sejong.ac.kr)
143-747 서울시 광진구 군자동 세종대학교 정보통신공학과
(전화: 02-3408-3695; 팩스: 02-3408-4330)

장르는 나라마다 혹은 사람마다 경계가 분명치 않고 문화, 가수, 시장에 따라 정의를 내리기 모호한 점이 있다 [1]. 음악 데이터 분석에 의한 자동 장르 분류는 효율적인 데이터관리와 음악 추천 등 다양한 어플리케이션에 적용이 가능하다. 또한, 수작업으로 분류를 하지 않아 경제적으로도 효율성이 있다. 음악 데이터 분석에 의한 장르 분류는 특징 벡터 추출, 분류기 등 다양한 방법으로 연구가 진행되고 발전하고 있다 [2-9].

음악 장르를 분류하기 위해선 음악 데이터를 분석하여야 한다. 음악 데이터는 박 (beat), 리듬, 박자 (Meter), 멜로디, 화음 (Chord) 등 다양한 분석 방법이 있다. 음악 데이터 분석 후에는 Gaussian Mixture Model (GMM)을 비

롯하여 Hidden Marcov Model (HMM), Nearest Neighbor (NN), K-Nearest Neighbor (KNN), Super Vector Machine (SVM)와 같은 다양한 분류기로 장르를 분류한다. Foote는 음악의 12차 Mel-Frequency Cepstral Coefficient (MFCC)의 히스토그램을 만들고, 분류기로 NN을 사용하여 장르를 분류하였다 [2]. Bagci는 13차의 MFCC와 델타 값을 구하고, Inter-Genre Similarity (IGS) 모델과 GMM 분류기를 사용하여 장르를 분류하였다 [3]. IGS 모델은 먼저 각 장르별 모델을 학습하고, 학습된 모델과 학습데이터를 이용하여 mismatch 되는 데이터를 만든다. 그 후, 각 프레임별로 IGS 모델과 각 장르별 모델의 확률을 측정하고, 각 장르별 모델에 대한 확률이 IGS 모델에 대한 확률보다 큰 프레임만 장르 분류에 사용한다 [3]. Tzanetakis는 10개의 음악 장르에 대하여 Spectral 특징 벡터로 Roll-off, Flux, Centroid, 그리고 MFCC 등 다양한 특징 벡터를 사용하였다 [4]. 분류기는 SVM을 사용하였다. Jiang은 음악의 Octave를 고려하여, MFCC와 다른 특징인 Octave-based Spectral Contrast (OSC)를 제안하여 장르 분류 성공률을 향상시켰다 [5]. 분류기로 GMM을 사용하여 Jazz, Pop, Romantic, Baroque, Rock의 5가지 장르에 대해서 약 82 % 장르 분류 성공률을 얻었다.

기존 음악 장르 분류 시스템은 MFCC를 이용하고 있다. 본 논문은 화자 인식에 사용하는 Decorrelate Filter Bank (DFB)를 장르 분류를 위한 특징 벡터로 사용하였다 [10-11]. 또한, MFCC, DFB 외에 Octave-based Spectral Contrast (OSC)를 멀티 특징 벡터로 이용하여 장르 분류 시스템을 구성하였다.

본 논문의 구성은 다음과 같다. II장에서는 음악 장르 분류 시스템에 대하여 설명하고, III장에서는 제안한 음악 장르 분류 시스템의 실험 결과를 보여준다. 마지막으로 IV장에서는 최종 결론을 내리도록 하겠다.

II. 장르 분류 시스템

장르 분류 시스템은 그림 1과 같이 특징 추출 (feature extraction), 모델링 (modeling), 분류 (classification) 과정으로 나눌 수 있다. 먼저 음악 데이터베이스에서 특징을 추출한다. 그 특징들을 모델링하여 모델을 생성한다. 그 후, 입력 음악에서 추출된 특징이 들어오면 생성된 모델을 이용한 분류기를 걸쳐 장르를 분류하게 된다. 음악의 특징 추출, 모델링 방법 그리고 분류 방법에 대한 자세한 내용은 다음과 같다.

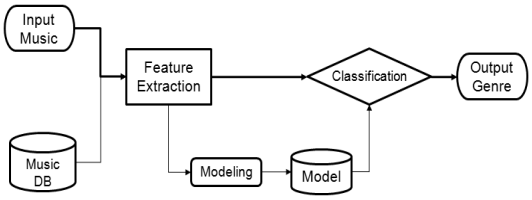


그림 1. 장르 분류 시스템의 블록도
Fig. 1. Block diagram of Genre classification system.

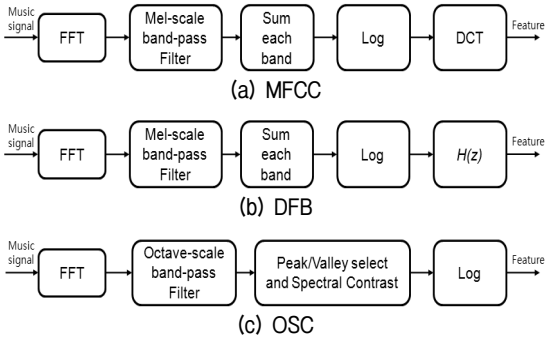


그림 2. 각 알고리즘별 특징 추출 블록도
Fig. 2. Block diagram of feature extraction for each algorithm.

2.1. 특징 추출

특징 추출 방법은 다양한 방법들이 있다. 본 논문에서는 기존 시스템에서 기본적으로 사용하는 MFCC 이외에 화자 인식에 주로 사용되는 DFB와 음악관련 특징으로 사용되는 OSC를 적용하였다.

2.1.1. MFCC

그림 2 (a)와 같이 MFCC는 시간 도메인에서 각 프레임에 Hamming window를 적용하여 구한 스펙트럼을 밴드별로 더하여 구한다.

한 프레임의 길이를 N이라고 할 때, Hamming window $w(n)$는 다음과 같다.

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right),\ 0 \le n \le N-1 \qquad (1)$$

윈도우가 씌워진 입력 신호는 Fast Fourier Transform (FFT)을 하고, 밴드가 B개인 Mel-scale band-pass filter로 스펙트럼을 스케일링 후 각 밴드별 스케일링된 값들을 더해준다. 이 값에 log를 적용하여 $LWS(b)(1 \le b \le B)$를 만든다. 마지막으로, 다음과 같은 Discrete Cosine Transform (DCT)을 적용하여 K차원의 MFCC 특징 벡터를 구한다.

$$MFCC(k) = \sum_{b=0}^{B-1} LWS(b)\cos\left(k\frac{\pi}{B}(b+0.5)\right),\ 0 \le k \le K \qquad (2)$$

2.1.2. DFB

MFCC는 기존 장르 분류에 주로 사용되는 특징이다. 하지만, [10-11]에서 제안된 화자 인식에 사용되는 DFB는 장르 분류에 아직 잘 사용되지 않고 있다.

DFB는 MFCC와 마찬가지로 각 밴드별 스케일링된 값들을 더한 후, log를 적용하여 $LWS(b)(1 \le b \le B)$를 만든다. 하지만, 그림 2 (b)와 같이 마지막 과정에서 DCT를 대신하여 FIR hipass-filter $H(z) = 1 - z^{-1}$를 통과하여, 다음과 같이 D 차원의 DFB 특징 벡터를 구한다.

$$DFB(d) = LWS(d+1) - LWS(d), \quad 0 \le d \le D < B \tag{3}$$

본 논문에서는 14개 Mel-scale band-pass filter를 사용하여 MFCC와 DFB를 추출하였다.

2.1.3. OSC

OSC는 각 밴드별 스펙트럼의 peak와 valley 값을 고려하여 측정된다. 대부분의 음악에서 강한 peak는 harmonic 부분과 연관되며, 강한 valley는 non-harmonic부분과 연관된다. 따라서, OSC는 밴드별 스펙트럼의 peak와 valley 값을 고려함으로써, 음악의 harmonic과 non-harmonic 성분을 고려할 수 있다 [5].

그림 2 (c)에 보듯이, OSC는 MFCC나 DFB와 다르게 mel-scale band-pass filter를 사용하지 않고, Octave-scale band-pass filter를 사용한다. 본 논문에서는 표 1과 같이 8개의 octave-scale band-pass filter를 이용하였다.

한 프레임의 길이와 FFT 포인트가 N으로 같을 때, 각 프레임에 대한 FFT 스펙트럼은 $\{x_1, x_2, \cdots, x_N\}$로 정의할 수 있다. i번째 밴드에 해당하는 FFT 포인트 수가 K_i이면, i번째 밴드의 스펙트럼은 $\{x_{i,1}, x_{i,2}, \cdots, x_{i,K_i}\}$와 같이 나타낼 수 있다. peak와 valley를 구하기 위해선 먼저 각 밴드별 스펙트럼을 내림차순으로 정리한다. 내림차순 정리된 i번째 밴드의 스펙트럼 $\{x'_{i,1}, x'_{i,2}, \cdots, x'_{i,K_i}\}$을 가지고, 아래 식을 이용하여 peak와 valley를 구할 수 있다.

표 1. Octave-scale band-pass filter의 주파수 범위
Table 1. Frequency range of octave-scale band-pass filter.

Band	Frequency (Hz)
1	[0~100)
2	[100~200)
3	[200~400)
4	[400~800)
5	[800~1600)
6	[1600~3200)
7	[3200~8000)
8	[8000~22050)

$$P_i = \log\left(\frac{1}{\alpha K_i}\sum_{j=1}^{\alpha K_i} x'_{i,j}\right) \tag{4}$$

$$V_i = \log\left(\frac{1}{\alpha K_i}\sum_{j=1}^{\alpha K_i} x'_{i,K_i-j+1}\right) \tag{5}$$

여기서 α는 상수로서 0.02부터 0.2까지 실험한 결과 α는 성능에 중요한 영향을 미치지 않았다 [5]. 따라서, 우리는 [5]와 같이 α를 0.02로 설정하여 사용하였다. 식 (4)와 (5)에서 구한 peak값과 valley값의 차이로 다음과 같이 spectral contrast를 구한다.

$$SC_i = P_i - V_i \tag{6}$$

OSC는 다음과 같이 I개 밴드에 대한 spectral contrast와 valley $\{SC_1, SC_2, \cdots, SC_I, V_1, V_2, \cdots, V_I\}$를 특징 벡터로 사용하게 된다. 본 논문에서는, 8개의 밴드를 사용하기 때문에 16차 OSC 특징 벡터를 사용하게 된다.

2.1.4. Texture window

화자 인식에서는 analysis window를 시간 도메인에 적용한 후 FFT 하여 추출한 특징 벡터를 독립적으로 사용한다. 하지만, 음악은 여러 개의 short-time 스펙트럼의 시리즈로 이루어져 있으므로, 그림 3과 같은 과정을 거쳐서 특징을 추출한다. 그림 3 (a)와 같은 입력 음악 신호에 대해서, 그림 3 (b)와 같은 window를 통해서 프레임별 신호를 얻고, FFT를 이용하여 주파수 도메인 신호를 얻는다. Texture window는 그림 3 (c)와 같이 복수개의 analysis window를 포함한다. Texture window는 analysis window에서 추출된 특징들의 평균과 분산을 계산하는데 이용한다. t번째 texture window의 N개의 analysis window에서 추출된 특징을 $\{F_{t,n}\}_{n=1,2,\ldots,N}$이라고 할 때, 평균과 분산은 다음과 같이 구한다.

$$M_t = \frac{1}{N}\sum_{n=1}^{N} F_{t,n} \tag{7}$$

$$V_t = \frac{1}{N}\sum_{n=1}^{N} (F_{t,n} - M_t)^2 \tag{8}$$

최종적으로, 하나의 음악 파일 내의 T개의 texture

window로 부터 구한 $\{M_t, V_t\}_{t=1,2,...,T}$을 이용하여 다음과 평균과 분산을 계산한 후, 이 값을 특징 벡터로 하여 장르를 분류한다.

$$\mu_M = \frac{1}{T}\sum_{t=1}^{T} M_t \tag{9}$$

$$\sigma_M = \frac{1}{T}\sum_{t=1}^{T} (M_t - \mu_M)^2 \tag{10}$$

$$\mu_V = \frac{1}{T}\sum_{t=1}^{T} V_t \tag{11}$$

$$\sigma_V = \frac{1}{T}\sum_{t=1}^{T} (V_t - \mu_V)^2 \tag{12}$$

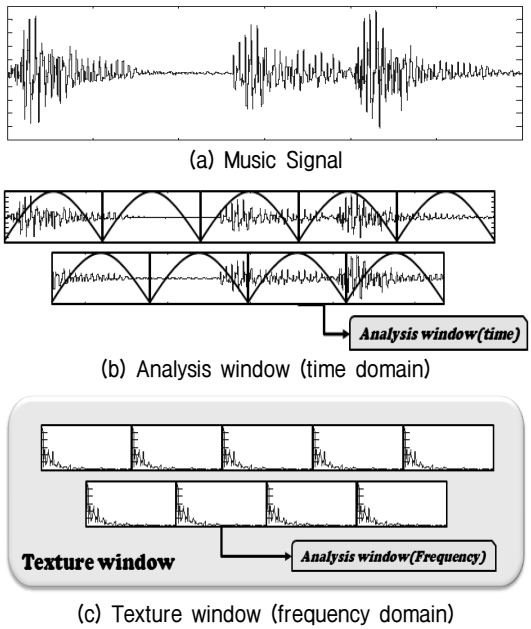
Analysis window가 K차 특징을 추출한다면, 각 texture window는 2*K 차 특징을 추출하며, 따라서 한 음악 파일에 대해서 총 4*K 차 특징을 사용하게 된다.

2.2. 장르 분류기

본 논문은 다양한 분류기 중 SVM을 사용하였다. SVM은 1979년 Vapnik와 AT&T Bell연구소에서 제안한 방법이다. 이 방법은 구조적인 위험 최소화 (Structural Risk Minimization)을 사용한다. 분류함수의 선이 데이터와 만날 때까지 확장한 폭을 마진 (Margin)이라고 하며, SVM은 이 마진이 최대가 되는 최적 분류 초평면 (Optimal Separating Hyperplane)을 다음과 같이 구한다.

$$f(x) = \sum_{i=1}^{n} \alpha_i \Xi(x_i, x) + b \tag{13}$$

여기서 α_i, $\Xi(x_i, x)$와 b는 각각 0이상의 값을 갖는 라그랑제 상수, 커널 함수, 그리고 바이어스를 나타낸다. 이 최적 분류 초평면과 만나는 데이터들을 Support Vector라고 부른다. 커널 함수로는 linear, polynomial, Radial Basis 함수 등이 있다 [12-13].



(a) Music Signal

(b) Analysis window (time domain)

(c) Texture window (frequency domain)

그림 3. Analysis window와 Texture window의 정의
Fig. 3. Definition of Analysis window and Texture window.

III. 실험 및 결과

음악 데이터베이스는 GTZAN을 사용하였다. GTZAN은 classical, country, disco, hiphop, jazz, rock, blues, reggae, pop, metal로 총 10개의 장르를 포함하며, 각 장르당 100곡, 한 곡당 30초로 16bit, 22050Hz, 모노, AU 파일 포맷으로 구성되어있다 [14].

본 실험에서는 기존 알고리즘으로서 Marsyas toolbox를 이용하여 성능을 비교하였다. Marsyas는 다양한 특징을 추출할 수 있고, 다양한 패턴 인식 방법을 이용해서 장르 분류를 할 수 있다.

표 2는 analysis window를 이용하여 추출한 특징 벡터들이다. Marsyas는 약 23 ms (512 samples)의 analysis window 별로 총 31차의 특징 벡터를 추출한다. 또한, 약 1 초의 texture window (20480 samples 또는 40개의 analysis window를 포함)에 대해서 1개의 analysis window만큼 overlap하면서 $\{M_t, V_t\}$을 구한 후, (9)-(12)을 이용해서 총 31*4 즉 124차의 특징벡터를 추출한다. [15]에 발표된 Marsyas 시스템의 기본 세팅에 따라

표 2. Analysis window를 이용하여 추출한 특징 벡터
Table 2. Extracted feature vectors using analysis window.

Feature vector	Feature vector Order	
	Marsyas	Proposed
Zero Crossing Rate	1	·
Spectral Centroid	1	·
Spectral Rolloff	1	·
Spectral Flux	1	·
Chroma	13	·
MFCC	14	13
DFB	·	13
OSC	·	16
Total	31	42

표 3. Marsyas와 제안한 알고리즘과의 성능 및 특징 벡터의 차수 비교 (texture window에서 추출한 특징 벡터들의 mean과 variance 값을 사용)
Table 3. Comparison in accuracy and feature-vector order between Marsyas and the proposed algorithm (using mean and variance values of extracted feature vectors in texture window).

Algorithm	Marsyas	Features (μ_M, σ_M, μ_V, σ_V)						
		MFCC	DFB	OSC	MFCC + DFB	MFCC + OSC	DFB + OSC	MFCC + DFB + OSC
Accuracy (%)	73.1	65.2	67.3	66	71.4	75.5	77.3	77.4
order	124	52	52	64	104	116	116	168

표 4. Marsyas와 제안한 알고리즘과의 성능 및 특징 벡터의 차수 비교 (texture window에서 추출한 특징 벡터들의 mean 값만 사용)
Table 4. Comparison in accuracy and feature-vector order between Marsyas and the proposed algorithm (using only mean values of extracted feature vectors in texture window).

Algorithm	Marsyas	Features (μ_M, μ_V)						
		MFCC	DFB	OSC	MFCC + DFB	MFCC + OSC	DFB + OSC	MFCC + DFB + OSC
Accuracy (%)	69.2	63.7	64	62.3	68.8	74.5	74.1	75.8
order	62	26	26	32	52	58	58	84

표 5. MFCC + DFB + OSC를 이용한 제안 알고리즘의 각 장르별 분류 성공률 (%)
Table 5. Genre-classification performance of the proposed algorithm with MFCC + DFB + OSC (%).

mean+variance (μ_M, σ_M, μ_V, σ_V)											mean (μ_M, μ_V)										
	bl	cl	co	di	hi	ja	me	po	re	ro		bl	cl	co	di	hi	ja	me	po	re	ro
bl	74	0	3	3	0	2	4	0	2	12	bl	79	0	3	2	0	3	5	0	0	8
cl	0	**95**	2	1	0	1	0	0	0	1	cl	1	**92**	2	0	0	1	0	1	0	3
co	0	0	76	4	0	2	0	2	2	14	co	1	0	73	4	0	4	0	3	3	12
di	2	1	4	70	5	0	1	3	4	10	di	4	1	3	70	5	0	1	6	2	8
hi	2	0	0	3	76	0	3	3	10	3	hi	2	0	0	4	74	0	3	2	12	3
ja	3	5	2	3	0	82	1	0	1	3	ja	4	6	6	2	0	78	1	0	2	1
me	2	0	0	1	0	0	90	0	1	6	me	2	0	0	3	0	0	89	0	1	5
po	0	1	6	2	2	0	0	83	2	4	po	0	1	7	4	1	1	0	81	2	3
re	2	0	2	4	15	1	1	3	67	5	re	3	0	4	5	14	2	1	3	64	4
ro	7	0	9	12	2	1	4	1	3	**61**	ro	10	0	6	12	1	1	4	3	5	**58**

※ blues (bl), classical (cl), country (co), disco (di), hiphop (hi), jazz (ja), metal (me), pop (po), reggae (re), rock (ro).

서, 본 논문에서도 1000개의 texture window를 사용하였다. 또한, 제안하는 특징벡터의 총 차수는 42차이며, 추출방법은 Marsyas와 동일하게 사용되었다.

장르 분류기로는 선형 커널 함수를 적용한 SVM을 사용하였다. 또한, 각 특징 벡터들을 정규화하여 장르를 분류하였다. 장르 당 90개의 파일을 학습 데이터로 쓰고, 10개를 장르 분류에 사용하여 총 10번의 cross-validation 실험을 하였다.

표 3은 식 (9)-(12)에서 구한 {μ_M, σ_M, μ_V, σ_V}를 모두 이용한 경우 Marsyas와 제안한 알고리즘의 성능 및 각 알고리즘 당 사용한 특징 벡터의 차수를 비교한 표이다. MFCC가 밴드 별 진폭 자체를 특징으로 사용함에 비교해서, DFB는 밴드 간 진폭의 차를 살펴보는데 보다 적합한 특징이다. 실험을 통해 관찰한 결과, 밴드 간 진폭의 변화도를 고려한 DFB가 MFCC보다 2.1 % 향상된 인식율을 보임을 알 수 있었다. 즉, 음악 장르 분류를 위해서는 밴드 자체의 진폭보다는 밴드 간 진폭의 차가 더 적합한 특징임을 알 수 있다. 특징 벡터 각각을 사용한 경우에는 DFB가 가장 우수하였고, OSC, MFCC 순으로 성능이 좋았다.

Marsyas는 분류 성공률이 73.1 %인 반면, 본 논문에서 제안한 DFB + OSC는 77.3 %로 4.2 %의 인식율 향상을 나타내었다. 특징 벡터 차수는 Marsyas보다 제안한 알고리즘이 8차의 이득을 보았다. MFCC + OSC의 경우 2.4 %의 성능이 증가하였으나 DFB + OSC보단 1.8 % 낮은 성능 결과를 보였다. 모든 특징 벡터의 조합인 MFCC + DFB + OSC를 사용할 경우, Marsyas보다 4.3 % 성능 향상이 있었다. 하지만 특징 벡터 차수에서 44차의 증

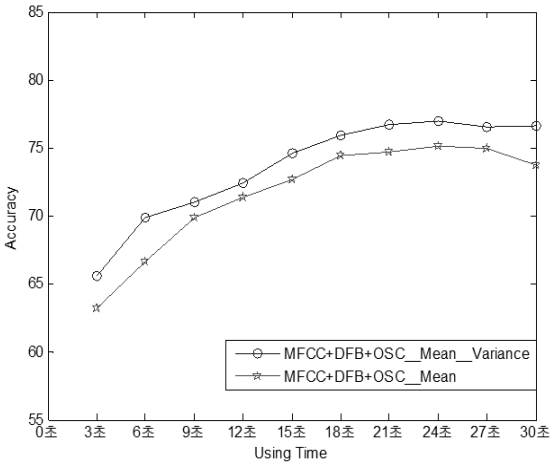


그림 4. 질의 (query)의 길이에 따른 장르 분류 결과
Fig. 4. Classification results with different query lengths.

가가 있었고, DFB + OSC와 성능차가 0.1 %로 거의 차이가 없었다.

표 4는 식 (9)-(12)에서 구한 $\{\mu_M, \sigma_M, \mu_V, \sigma_V\}$를 모두 이용하지 않고, $\{\mu_M, \mu_V\}$만을 사용하여 특징 벡터의 차수를 줄이는 실험을 한 결과이다. 표 4의 MFCC + DFB + OSC는 75.8 %로 표 3의 MFCC + DFB + OSC보다 1.6 % 분류 성공률이 낮지만, 특징 벡터의 차수에서 절반인 84차의 이득을 볼 수 있었다. 표 3과 표 4에서 분류 성공률 대비 특징 벡터 차수를 비교해본 결과, 특징 벡터 차수가 많이 줄어 들어도 성능이 많이 저하 되지 않음을 알 수 있었다.

표 5는 표 3과 표 4에서 MFCC + DFB + OSC를 이용한 경우의 각 장르별 분류 성공률을 나타낸다. 각 장르별 테스트로 10개의 파일을 쉬프트하면서 10번의 실험으로 총 100개의 파일을 분류하였다. Classical music이 가장 분류 성공률이 뛰어나며, rock music이 가장 분류 성공률이 저조하였다. Classical music은 저주파에 대부분의 에너지가 집중되어 있어서 분류 성공률이 뛰어났다. 또한, metal music은 다른 장르에 비해서 고주파 에너지가 높은 편이어서 분류가 쉬운 편이었다. 반면, rock이나 기타 장르들은 저주파에서 고주파까지 고루 에너지가 분포하여 분류에 어려움이 있었다. 특히, rock music은 blues, country, disco music 들로 오분류 되는 경우가 많이 있었다.

그림 4에서는 총 길이가 30초인 음악 데이터에 대하여 질의 (query) 길이별 분류 성능을 테스트하였다. 즉, 질의 길이를 3초씩 증가시키면서 각 질의 길이에 따른 장르 분류 성공률을 보여준다. 예를 들어, 질의 길이가 3초인 경우에는 총 30초 중 처음 3초를 이용하여 학습 및 분류를 시행하였다. 질의의 길이가 길어질수록 학습 데이터와 테스트 데이터가 증가하여 분류 성공률도 증가하였다. 즉, 안정적으로 장르를 분류하기 위해서는 21초 이상의 데이터를 이용해야 하는 것을 보여준다. 또한, $\{\mu_M, \sigma_M, \mu_V, \sigma_V\}$를 모두 이용한 경우가 $\{\mu_M, \mu_V\}$만을 이용한 경우보다 성능이 우수함을 알 수 있었다.

IV. 결 론

본 논문에서는 DFB와 OSC 특징 벡터를 결합하여 장르 분류 시스템의 성능을 개선하였다. DFB만을 사용하여 장르를 분류한 결과, MFCC와 OSC보다 개선된 성능을 보였다. OSC와 복합 특징 벡터로 구성할 경우 MFCC보다는 DFB의 경우가 성능이 더 좋은 것을 확인하였다. 또한, Marsyas 시스템보다 4.2 %의 높은 성능 향상과 8차의 특징 벡터 차수 이득을 얻을 수 있었다. MFCC + DFB + OSC는 DFB + OSC 시스템보다 52차만큼 특징 벡터의 차수 증가가 있었지만 0.1 %의 근소한 성능 개선만을 얻을 수 있었다.

음악 데이터의 질이 길이를 많이 볼수록 성능이 증가하지만, 일정 이상을 볼 경우 성능의 큰 차이가 없었다. 특징 벡터의 차수를 감소한 경우 급격한 성능 저하를 보이지 않았고, 차수를 증가시켜도 성능이 항상 향상되지 않았다. 즉, 장르 분류 성능에는 특징 벡터의 차수보다 특징 벡터의 종류와 조합이 더 많은 영향을 주는 것을 확인하였다. 향후에는 장르 분류에 보다 적합한 특징 벡터와 분류기를 제안할 예정이다.

감사의 글

이 논문은 2010년도 정부 (지식경제부)의 재원으로 산업원천기술개발사업의 지원을 받아 수행된 연구임 (No. 10037244).

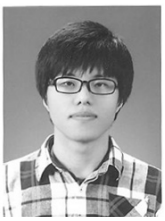
참 고 문 헌

1. N. Scaringella, G. Zoia, and D. Mlynek, "Automatic genre classification of music content: A survey," *IEEE Signal Process.*, vol. 23, no. 2, pp. 133-141, 2006.
2. J. Foote, "Content-based retrieval of music and audio," in *Proc. SPIE Multimedia Storage Archiving Systems II*, vol. 3229, pp. 138-147, 1997.
3. U. Bagci and E. Erzin, "Automatic Classification of musical Genres using Inter-genre Similarity," *IEEE Signal process. Letters*, vol. 14, no 8, pp. 521-424, 2007.

4. G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 293–302, 2002.
5. D. N. Jiang, L. Lu, H. J. Zhang, J. H. Tao, and L. H. Cai, "Music type classification by spectral contrast feature," in *Proc. IEEE Int. Conf. Multimedia and Expo,* vol. 1, pp. 113–116, 2002.
6. Y. Zhang and J. Zhou, "A study on content-based music classification," in *Proc. Signal Process. and Its applications*, vol. 2, pp. 113–116, 2003.
7. S. Z. Li, "Content-based classification and retrieval audio using the nearest feature line method," *IEEE Trans. Speech and Audio Process.*, vol. 8, no. 5, pp. 619–625, 2000.
8. C. Xu, N. C. Maddage, and X, Shao, "Automatic music classification and summarization," *IEEE Trans. Speech and Audio Process.*, vol. 13, no. 3, pp. 441–450, 2005.
9. 박철의, 박만수, 김성탁, 김회린, "피치 히스토그램과 MFCC-VQ 동적 패턴을 사용한 음악검색," *한국음향학회지*, 24권, 3호, 178–185쪽, 2005.
10. J. Ming, T. J. Hazen, J. R. Glass, and D. A. Reynolds, "Robust speaker recognition in noisy conditions," *IEEE Trans. Audio, Speech, Language Process.*, vol. 15, no. 5, pp. 1711–1723, 2007.
11. J. Jung, K. Kim, and M. Y. Kim, "Noise robust speaker identification based on the advanced missing feature theory," *Electronics Letters*, vol. 46, no. 14, pp. 1027–1029, 2010.
12. Y. Wang, "A Tree-Based Multi-class SVM Classifier for Digital Library Document," in *Proc. IEEE Int. Conf. Multimedia and Information Technology*, pp. 15–18, 2008.
13. 한학용, "패턴인식 개론 : MATLAB 실습을 통한 입체적 학습," 2005.
14. GTZAN Genre Collection Database, "http://marsyas.info/download/data_sets"
15. G. Tzanetakis, "MARSYAS SUBMISSIONS TO MIREX 2009," *Music Information Retrieval Evaluation eXchange* (*MIREX*), 2009.

저자 약력

•임 신 철 (Shin-Cheol Lim)



2005년 3월 ~ 2011년 2월: 세종대학교 정보통신공학과, 공학사
2011년 3월 ~ 현재: 세종대학교 정보통신공학과, 석사 과정
※ 관심분야: 화자 인식, 음악정보검색, Compressed sensing

•장 세 진 (Sei-Jin Jang)

1995년 2월: 경북대학교 전자공학과, 공학사
1997년 2월: 경북대학교 대학원, 전자공학과, 공학석사
1997년 ~ 2002년: 대우전자 전략기술연구소, 전임연구원
2002년 ~ 현재: KETI 차세대음향산업지원센터, 센터장

•이 석 필 (Seok-Pil Lee)

1990년 2월: 연세대학교 전기공학과, 공학사
1992년 2월: 연세대학교 대학원, 전기공학과, 공학석사
1997년 2월: 연세대학교 대학원, 전기전자공학과, 공학박사
1997년 ~ 2002년: 대우전자 영상 연구소, 선임연구원
2002년 ~ 현재: KETI 디지털미디어연구센터, 센터장

•김 무 영 (Moo Young Kim)

1989년 3월 ~ 1993년 2월: 연세대학교 전자공학과, 학사
1993년 3월 ~ 1995년 2월: 연세대학교 전자공학과, 석사
1995년 2월 ~ 2000년: 삼성종합기술원 전문연구원
2001년 1월 ~ 2004년 11월: Royal Institute of Technology (KTH, 스웨덴) Dept. Signals, Sensors, Systems, 박사
2004년 ~ 2005년 2월: Royal Institute of Technology (KTH, 스웨덴) Dept. Signals, Sensors, Systems, PostDoc
2005년 2월 ~ 2006년 8월: Ericsson Research (스웨덴), Senior Research Engineer
2006년 8월 ~ 현재: 세종대학교 정보통신공학과, 부교수
※ 관심분야: 음성/오디오 신호처리 및 코딩, 패턴인식, 정보이론