실용 QbSH 시스템 개발에 관한 연구

송재종*, 장세진*, 박호종**
*전자부품연구원 디지털미디어 연구센터
**광운대학교 전자공학과
e-mail:jcsong@keti.re.kr

A Study of Practicla QbSH System Development

Chai-Jong Song*, Sei-Jin Jang*, Hochong Park**
*DigitalMedia Research Center, KETI
**Dept of Electronics Engineering, Kwangwoon University

요 약

본 논문에서는 다성 음원 기반의 다양한 응용 분야에서 적용이 가능한 실용 Query by Singing/Humming (QbSH) 시스템을 구현한다. 이를 위해서 검색 서버을 위한 소프트웨어 스택을 개발하고 다양한 형태의 클라이언트와 함께 임베디드 시스템과 같은 저성능 클라이언트를 위한 허밍검색 에이전트를 개발하여 기존의 클라이언트 시스템에 플러그인 할 수 있는 모듈을 개발한다. 실용 QbSH 시스템을 개발하기 위하여 각각의 핵심 알고리즘들의 최적의 조합을 이루어 통합된 전체시스템의 성능을 평가한다. 학술적인 연구에만 그쳤던 허밍기반 오디오 검색 서비스를 상업적으로 이용하기 위해 다양한 응용분야의 프로토타입을 구현한다.

1. 서론

방송통신융합서비스의 확산과 함께 사용자들이 접하는 콘텐츠의 종류와 데이터 량이 급격히 증가하고 있다. 웹과 인터넷을 기반으로 하던 정보통신기술은 모바일 기술의 급속한 발전에 힘입어 데스크톱이라는 공간적인 한계를 벗어나, 다양한 무선 네트워크를 기반으로 하는 Social Network Service (SNS)로 빠르게 진화하고 있다. 급격하게 변화하는 미디어 서비스 환경을 보다 효율적이고 편리하게 이용하기 위한 방법들이 제시되고 있으며, 방대한 양의 데이터에서 필요한 정보를 편리하고 빠르게 검색할 수 있는 방법들이 주요 이슈가 되고 있다. 이러한 변화를 이끌고 있는 Google, Amazon과 같은 다국적 기업들은 멀티미디어 콘텐츠 검색을 전략적 핵심 기술로 삼고 관련 기술의 연구개발에 심혈을 기울이고 있다. 미디어 검색 기술은 메타데이터 기반의 질의방식이 주를 이루었으나 최근에는 질의의 간편성과 제한성을 충족시키기 위한 내용기반 질의 방식의 요구가 급증하고 있다. 이러한 요구는 스마트폰의 등장으로 Shazam, SoundHound, 네이버 음악검색, 다음 음악검색과 같은 내용 기반 음악검색 서비스들을 출현시켰다. 하지만 위 서비스들은 검색하고자 하는 음악의 원본 샘플이 필요하다는 단점을 가진다. 이러한 단점을 극복하고 메타데이터 및 원본 샘플의 정보 없이 원하는 음악을 검색할 수 있는 방법이 Query by Singing/Humming (QbSH)이다. 다른 검색기술들과 같이 QbSH의 기본 기능은 원하는 음악을 Database (DB)로부터 정확하고 빠르게 찾는 것이다. 이를 위하여 지금까지의 QbSH 연구는 Musical Instrument Digital Interface (MIDI)와 같은 단성음원을 사용하였다. 여기에는 여러 가지 이점이 있겠지만 검색의 정확도 및 검색 시간의 단축에 대한 것이 가장 크다. 이에 대한 대표적인 방법이 Ghias에 의해 제안된 Up-Down-Repeat (UDR) 문자열을 사용하는 것이다. 그는 연속되는 피치 값을 비교하여 앞의 값보다 높음을 나타내는 U, 낮음을 나타내는 D, 반복을 나타내는 R의 3가지 문자로 특징을 표시하였다 [10]. 이후 UDR 방식을 기반으로 한 MELDEX system, Themefiner system 등이 제안되었으나 만족할 만한 성능향상을 보여주진 못했다 [11][12]. Roger Jang은 Music Information Retrival Evaluation eXchange (MIREX)의 QbSH task를 위한 평가데이터 집단을 제공하고 기존의 UDR 방식을 사용하지 않고 피치 값을 세미톤으로 표현함과 동시에 Dynamic Time Warp (DTW)와 Linear Scaling (LS) 기반의 검색 시스템을 제안함으로써 성능을 향상시켰다[8][9].

이러한 연구를 기반으로 Midomi는 상용 QbSH 서비스를 선보이면서 사용자들의 관심을 끌기도 했다. 이 시스템은 허밍을 이용하여 특징 데이터베이스를 만들기 때문에 데이터의 집중현상의 문제점을 가진다. 이는 유명한 곡이나 사용자가 자주하는 허밍 질의에 대한 노래의 검색 정확도는 높일 수 있지만 그렇지 않는 노래들에 대해서는 검색 결과를 제공할 수 없다. 이러한 단점을 극복하고 상용 서비스를 제공하기 위해서는 반드시 음악의 원곡에서 특징 데이터를 추출하고 이를 기반으로 하는 데이터베이스를 구축해야한다. 다성 음원을 사용함으로써 데이터 집

중현상을 피할 수 있을 뿐만 아니라 새롭게 발매되는 음악에 대한 특징 DB를 쉽게 구성할 수 있다.

본 논문에서는 다성 음원에서 특징을 추출하여 생성한 DB를 기반으로 실용 QbSH 시스템을 제안하고 개발한다. MIDI나 허밍과 같은 단성음원을 사용하여 특징 DB를 구성하는 경우는 검색엔진의 성능이 전체 시스템의 성능을 좌우하게 된다. 하지만 다성 음원을 이용하여 특징 DB를 구성하는 경우는 전체 시스템의 성능이 특징 추출 알고리즘의 정확성과 매칭 알고리즘의 정확성에 함께 영향을 받게 된다. 또한, 다성 음원에서 특징을 추출하기 때문에 추출된 특징에는 오류가 다 수 포함 될 수 있고 이는 매칭 알고리즘에 다양한 형태로 영향을 미치게 된다. 전체 성능을 높이기 위해서는 각 알고리즘의 성능도 중요하지만 알고리즘 사이의 역학관계를 이해하고 통합하는 방법 또한 매우 중요하다. 본 논문은 다양한 실험을 통하여 최적의 성능을 위한 시스템을 개발한다.

2. 실용 QbSH 개발 및 구현

제안하는 실용 QbSH 시스템은 그림 1와 같이 네 개의 핵심 부분으로 구성된다. 첫 번째는 사용자 허밍 질의 신호에서 배경잡음을 억제하고 특징을 추출하는 부분, 두 번째는 다성 음원에서 보컬 신호를 강화하고 특징을 추출하는 부분, 세 번째는 허밍 질의 시퀀스와 특징 DB 시퀀스 사이의 유사도를 기반으로 하는 검색 엔진과 그 결과를 추천하는 부분, 그리고 마지막으로 추출된 특징 시퀀스를 기반으로 한 특징 DB 색인 부분이다. 우리는 실용 QbSH 시스템을 위한 핵심 알고리즘들을 [4-7]을 통하여 제안하였다. 본 논문에서는 위 알고리즘을 이용하여 실용적인 QbSH 시스템을 개발한다.

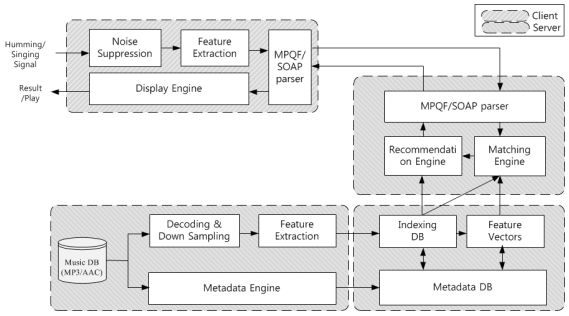

그림 1. 실용 QbSH 시스템 구성도

QbSH 시스템은 기본적으로 서버와 클라이언트로 구성된다. 서버는 QbSH 시스템에서 클라이언트 접속 관리, 질의에 대한 검색, 시스템의 상태에 대한 업데이트와 같은 거의 모든 작업을 수행하기 때문에 효과적인 관리가 필수적이다. 이를 위하여 우리는 그림 2과 같이 서버를 위한 소프트웨어 스택을 설계 및 구현한다. 소프트웨어 스택의 주요 요소는 음원, 특징벡터, 색인, 메타데이터를 위한 4개의 DB들과 아래와 같은 기능을 제공하는 6개의 관리자들이다.

- **Connection Manager** : 클라이언트의 접속을 받아들이고 관리한다. 관리자는 접속된 클라이언트들에게 각각의 식별 번호를 부여하고 다른 관리자들에게 새로운 클라이언트가 접속했음을 식별 번호와 함께 알린다. 이후 모든 관리자들은 통보 받은 식별번호를 바탕으로 모든 작업을 수행한다. 식별번호는 접속된 순으로 64 비트를 이용하여 0에서 시작하여 1씩 증가하게 되고 범위를 넘어서면 새롭게 시작한다. 접속관리자는 클라이언트의 연결요청, 연결해제 요청, 네트워크 상태, 예상치 못한 연결 해제, 메시지 송수신, 접속 다중 쓰레드 관리 등을 수행한다. 또한 접속보안과 관련된 업무를 함께 포함한다. 다양한 네트워크 프로토콜 및 스트리밍 서비스를 지원하기 위하여 관련 모듈의 플러그인으로 지원한다.
- **Update & Feature Manager** : 새로운 음원인 MP3나 Advanced Audio Coder (AAC) 파일의 추가로 데이터베이스 변경이 필요한 경우 특징관리자와 색인관리자에게 이를 알리고 특징 벡터, 색인, 메타데이터 DB를 업데이트 한다. 음원이 아닌 특징벡터가 추가되는 경우 업데이트 관리자는 색인 관리자에게만 알리고 특징 벡터와 색인DB을 업데이트한다. 이 경우 메타데이터 및 음원 DB를 서버관리자가 직접 업데이트 할 수 있도록 관련 API를 제공한다. 특징 추출기와 보컬 강화를 위한 전처리 모듈을 분리하여 플러그인으로 지원한다. 이는 보컬 강화을 위한 전처리기가 특징추출기에 종속되지 않기 때문에 다양한 전처리기를 지원하기 위함이다.
- **Search & Index Manager** : 검색엔진의 플러그인, 플러그아웃을 지원하며 클라이언트로부터 들어온 허밍 질의를 검색엔진에게 넘겨주고 색인 관리자에게 새로운 질의가 들어왔음을 알린다. 색인관리자는 새로운 질의가 있음을 통보받으면 색인 정보를 검색엔진에게 넘겨준다. 검색엔진은 넘겨받은 허밍 질의와 색인 정보를 이용해 유사도를 판별한다. 색인관리자는 새로운 음원이나 특징벡터가 추가되었음을 통보 받게 되면 색인 엔진을 호출하여 색인 및 메타데이터 DB을 업데이트한다.
- **Storage Manager** : 음원, 특징벡터, 색인, 메타데이터 DB를 관리하고 상위 관리자들의 요구에 빠른 응답을 제공해야한다. 특히 색인 DB는 검색 속도와 직접적인 관련이 있기 때문에 빠른 응답은 필수적이다. 이를 위하여 하나의 그룹에 속해있는 시퀀스데이터는 연속된 디스크 페이지에 저장하고 각 그룹에 대한 페이지 ID와 그룹의 색인 정보를 쌍으로 하는 엔트리를 생성하여 VA-파일에 저장한다 [1].

소프트웨어 스택은 검색, 색인, 특징 추출기와 같은 지속적인 성능향상이 필요한 기능들을 언제든지 교체 및 업그레이드가 가능할 수 있도록 관련 관리자들의 플러그인 형식을 지원한다. 엔진 개발자들은 플러그인 형식에 맞추어 독립적으로 각각의 엔진을 개발할 수 있다. 최상위 계층에 프로그램 개발자를 위한 Application Programming

Interface (API)를 제공한다. 미디어 변환기와 스트리밍 엔진에 대하여 간략하게 기술한다.

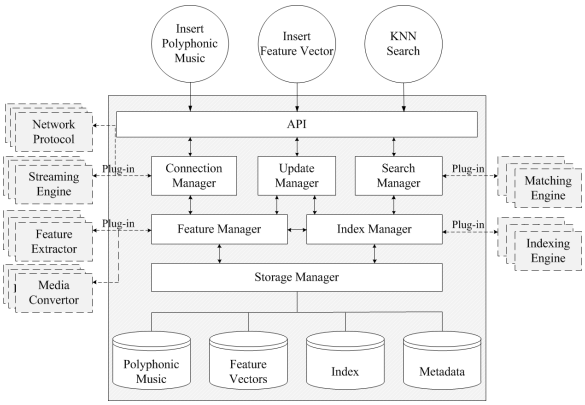

그림 2. 서버 소프트웨어 스택 구조도

- **Media Convertor** : 특징 추출기의 입력 신호 형식은 8 kHz 표본주파수를 가지는 모노채널의 16 비트 오디오 신호이다. 미디어 변환기는 다양한 형식의 다성 음원 신호를 특징 추출기의 입력신호로 변환해 주는 역할을 한다. 미디어 변환기는 MP3, AAC 디코더를 포함하고 있으며 스테레오 채널을 모노채널로 변환하는 다운믹싱과 표본주파수를 8 kHz로 변환해 주는 재 표본화 모듈을 포함한다. 다운믹싱은 일반적으로 사용되는 좌위 채널인 L 과 R의 합의 절반 값을 취하는 방법을 사용한다. 음악 신호인 경우 L 과 R의 차이가 크지 않기 때문에 위와 같은 방법을 사용해도 시스템에 미치는 영향은 미미하다. 반면 재 표본화 과정에서 저역 통과 필터를 이용한 업/다운 표본화가 필수적이고, 이는 특징 벡터 추출 과정에 상당한 영향을 미치게 된다. 본 논문에서는 창 함수를 이용한 저역 통과 필터를 구현하고 실험을 하였다. Hamming, Hanning, Blackman, Bartlett, Kaiser 창 함수를 대상으로 실험을 진행하였고, Kaiser 창 함수가 개발된 특징 추출기와 가장 적합함을 알 수 있었다.
- **Network Protocol & Streaming Engine** : 다양한 클라이언트 플랫폼을 지원하기 위하여 허밍 질의 및 검색 결과를 MPQF 표준을 이용하여 형식화하고, Simple Object Access Protocol (SOAP) 혹은 Transmission Control Protocol/Internet Protocol (TCP/IP)의 소켓을 이용하여 전송한다. 검색 결과에 대한 스트리밍 서비스를 제공하기 위하여 RTP/RTSP기반의 실시간 RTSP 스트리밍 엔진을 구현하였다. RTSP 스트리밍 서비스 기능을 가지고 있지 않은 클라이언트를 위하여 다운로드 서비스를 제공한다. RTSP 스트리밍과 다운로드 서비스는 허밍 질의와 가장 유사한 부분을 찾아서 30초 분량을 제공한다. 이는 사용자의 허밍 질의와 같은 부분을 들려줌으로써 검색결과가 사용자가 원하는 노래인지를 쉽게 판별 할 수 있을 뿐만 아니라 원음에 대한 저작권을 보호하기 위함이다.

서버는 시작과 함께 모든 DB를 초기화하고 DB 관리 및 업데이트를 위한 새로운 쓰레드를 생성한다. 음원 DB에 새로운 다성 음원이 추가 되었으면 음원으로부터 특징을 추출하고 관련 DB을 업데이트한다. 새로운 클라이언트의 접속요구가 있는지를 확인하여 새로운 검색 쓰레드를 생성하여 클라이언트의 질의에 대한 서비스를 제공한다. 소프트웨어 스택은 동시 다중 접속을 지원하기 위하여 다중 쓰레드 구조로 설계되었다. 서버를 중지하지 않고도 새로운 음원을 추가 할 수 있도록 하기 위하여 DB 쓰레드를 생성하여 백그라운드로 새로운 음원이 추가 되는지를 감지한다. 이때, 새로운 음원의 추가를 폴링방식을 사용하게 되면 검색 시스템의 성능에 영향을 줄 수 있기 때문에 음원추가 알람방식을 사용한다. 모든 클라이언트가 접속해제 되고 서버 운영자가 서버의 사용중지를 요구하게 되면 모든 쓰레드가 종료되는 것을 확인한 후 서버는 종료된다. 그림 3은 응용모델을 위해 개발된 프로토타입이다.

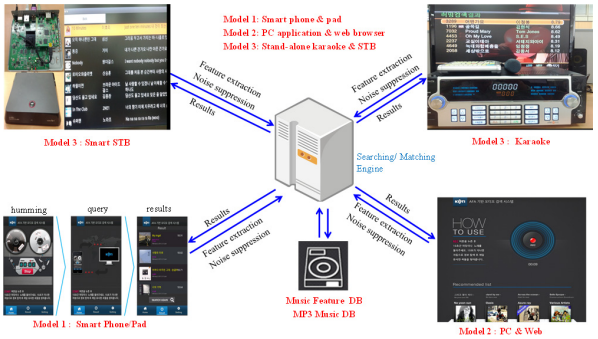

그림 3. 실용 QbSH 시스템을 이용한 다양한 응용 모델

3. 실험 및 평가

본 논문에서는 시스템의 개발 및 객관적 성능 테스트를 목적으로 AFA450 과 AFA2000으로 명명한 세 가지 데이터 집합을 새롭게 구성한다. AFA450에는 다성 음원 450곡에 대한 MP3/AAC파일과 MIDI파일로 구성되고, AFA2000은 AFA450을 포함한 2,000곡으로 구성된다. 각 노래의 재생시간은 동요와 캐럴을 제외한 모든 노래는 4분 이상, 6분 이내이다. 데이터 편중을 피하기 위해 대상 데이터 집단을 8가지 장르로 구분하고 장르별 비율을 고려하여 구성하였다.

서버의 검색 성능 테스트를 위하여 허밍 질의 데이터 집합을 사용하고, 각각 8, 10, 12초의 허밍질의에 대한 20개의 매칭결과를 기반으로 한 Mean Reciprocal Rank (MRR)을 사용한다. MRR은 질의 시스템의 성능을 측정하기 위한 방법으로 광범위하게 사용되고 있다 [9]. 검색 시스템의 정답 비율을 알아보기 위해 각각의 질의에 대한 검색의 정답이 상위 1위, 5위, 10위, 20위내에 포함되는 비율을 함께 측정한다. 그림 4는 검색 시스템의 성능을 MRR과 정답 비율로 나타낸다. 우리는 입력 질의의 길이

가 10초 이상에서 좋은 성능을 나타내는 것을 관찰할 수 있고 상위 5위내에 거의 모든 매칭결과가 존재함을 알 수 있다. 이 결과는 검색 대상 데이터 집합인 AFA450이나 AFA2000에서 모두 비슷한 결과를 얻을 수 있음을 알 수 있다. 또한 한곡의 질의에 대한 평균 검색시간은 AFA450에서는 평균 2.9초이며, AFA2000에서는 평균 8.6초이다.

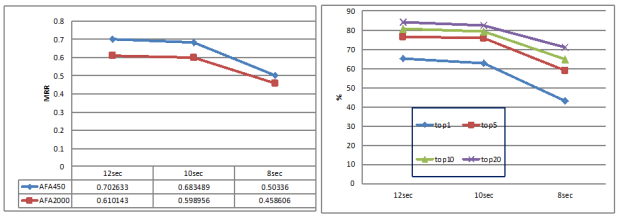

그림 4. 시스템 성능: 좌)MRR, 위)AFA450에 대한 랭킹비율

4. 결론

우리는 실용 QbSH 서비스를 위한 시스템을 개발하였다. 이를 위하여 허밍 질의 데이터 집합과 세 가지의 음원 검색 데이터 집합을 구성하였다. 구성된 데이터 집합을 이용하여 개발된 시스템의 성능을 검증하였다. QbSH 시스템의 핵심인 검색 서버 개발을 위한 소프트웨어 스택을 설계 및 구현하였다. 소프트웨어 스택은 여섯 개의 관리자와 네 개의 데이터베이스로 구성되었고 새로운 성능 개선을 쉽게 하기 위하여 모듈 별로 플러그인 방식을 지원한다. 다양한 분야의 응용 서비스를 지원하기 위하여 세 가지의 응용 시나리오를 정의하고 각각에 알맞은 형태의 클라이언트를 개발하였다. 특히 임베디드 시스템을 위한 DSP 모듈은 기존 제품의 변경 없이 새로운 QbSH 서비스를 제공할 수 있도록 하였다. 위와 같은 다양한 모델의 서비스 시나리오에 해당하는 프로토타입을 개발함으로써 사용자들에게 어떠한 환경에서도 QbSH서비스를 이용할 수 있도록 통합솔루션을 제공한다.

참고문헌

[1] R. Weber, H.-J. Schek and S. Blott. “A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces,” Proceedings of VLDB, pp. 194 - 205, 1998.

[2] A. P. Klapuri, "Multiple fundamental frequency estimation based on summing harmonic amplitudes," in Proc. Int. Conf. Music Information Retrieval, pp. 216-221, 2006

[3] Kichul Kim, K.R Park, S.J Park, S.P Lee and M.Y Kim. "Robust Query-by-Singing/Huming System against Background Noise Environments," IEEE Trans. on Consumer Electronics, vol.57, no.2, pp. 720-725, May 2011

[4] 이세원, 송재종, 이석필, 박호종, “보컬 피치 검출의 성능향상을 위한 보컬 강화기술,” 한국음향학회지, 제30권, 제6호, pp. 353-359, 2011

[5] 윤제열, 이석필, 서경학, 박호종, “하모닉 구조를 이용한 다성 음악의 주요 멜로디 검출” 전자공학회지, 제47권 SP편 제5호, pp. 109-115, 2010

[6] Dalwon Jang, Chai-Jong Song, Saim Shin, Sung-Joo Park, Sei-Jin Jang, and Seok-Pil Lee, "Implementation of a matching engine for a practical query-by-singing/humming system," Proc. ISSPIT, Dec. 2011

[7] 길기정, 송석일, 송재종, 이석필, 장세진, 이종설 “최소 DTW 거리 기반의 데이터 시퀀스 색인 기법,” 한국콘텐츠학회지, vol. 11, no. 12, pp. 52-59, 2011

[8] J. S. R. Jang and M. Y. Gao, " A query-by-singing system based on dynamic programming," in Proc. Int. Workshop on Intelligent Systems Resolution, pp. 85-89, 2000

[9] J. S. Downie, “The music information retrieval evaluation exchange(2005-2007): A window into music information retrieval research”, Acoust. Sci. & tech, 29, 4, 2008

[10] A. Ghias, J. Logan, D. Chamberlin, and B.C. Smith, "Query by Humming: Musical Information Retrieval in an Audio Database," in Proc. of the third ACM Int. Conf. on Multimedia, pp. 231-236, 1995

[11] R. J. McNab, L. A. Smith, I. H. Witten, C. L. Henderson, and S. J. Cunningham, "Towards the digital music library: Tune retrieval from acoustic input," in Proc. First ACM Conf. on Digital Libraries, pp. 11-18, Bethesda, MD, USA, 1996

[12] A. kornstadt, "Themefinder: a web-based melodic search tool," Computing in Musicology, vol. 11, pp. 231-236, 1998