

# **대화형 사용자 인터페이스 개론 (GITA370)**

**Chapter 5, 6, 7**

**김 지 환**

**서강대학교 정보통신대학원**

**Email: kimjihwan@sogang.ac.kr**

# 목차

## 5. 한국어 음성인식 구현에 있어서의 이슈

### 5.1 발음열 변환 (Grapheme to Phoneme (G2P))

## 6. 개발 툴 및 음성 코퍼스 소개

### 6.1 휴대폰용 음성인식 앱

- 현 음성인식 기술의 수준을 알 수 있음

### 6.2 음성인식기 I/O 라이브러리

- 개발된 음향모델/언어모델에 대한 I/O 라이브러리. 모델을 학습하지 않음

### 6.3 음성인식기 개발용 툴 및 음성 코퍼스

#### 6.3.1 음성인식기 개발 툴

#### 6.3.2 언어모델 개발 툴

#### 6.3.3 음성 코퍼스

# 목차

## 7.음성외 주요 오디오 정보 처리

### 7.1.1음악인식 소개

#### 7.1.1.1음악검색 서비스

#### 7.1.1.2 음악인식 알고리즘

### 7.1.2 Audio fingerprint 음악인식 알고리즘 구현

#### 7.1.2.1 Gracenote (Mobile MusicID) 구현

#### 7.1.2.1 Shazam (Shazam encore)구현

### 7.1.3 QBH(Query by Humming) 방식의 음악인식 알고리즘

### 7.2.1 오디오 이벤트 분류의 필요성

#### 7.2.2.1 기존 오디오 이벤트 분류

#### 7.2.2.2 DNN 기반 오디오 이벤트 분류

#### 7.2.2.3 CNN 기반 오디오 이벤트 분류

## 5. 한국어 음성인식 구현에 있어서의 이슈

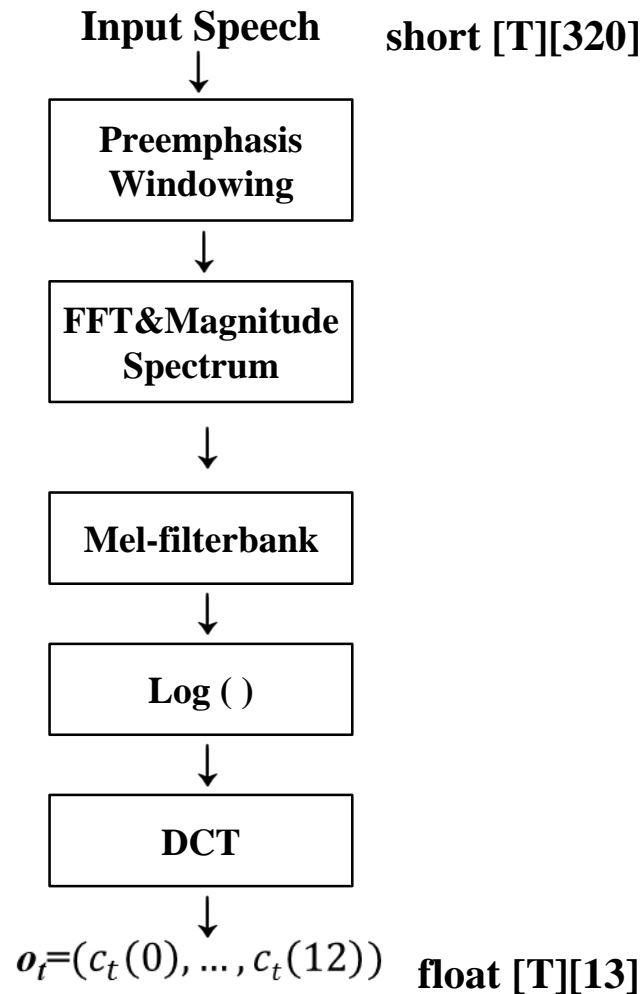
### ■ 음성인식 문제정의

$$\underbrace{arg_w max}_{\text{디코딩}} \underbrace{P(O|W)}_{\text{음향 모델}} \underbrace{P(W)}_{\text{언어 모델}}$$

- 따라서, 음성인식 시스템 핵심 컴포넌트는
  - 음향 모델  $P(O|W)$
  - 언어 모델  $P(W)$
  - 디코딩 네트워크( $arg_w max$ )
- 어휘(인식 가능한 단어 set) 의 구현이다.

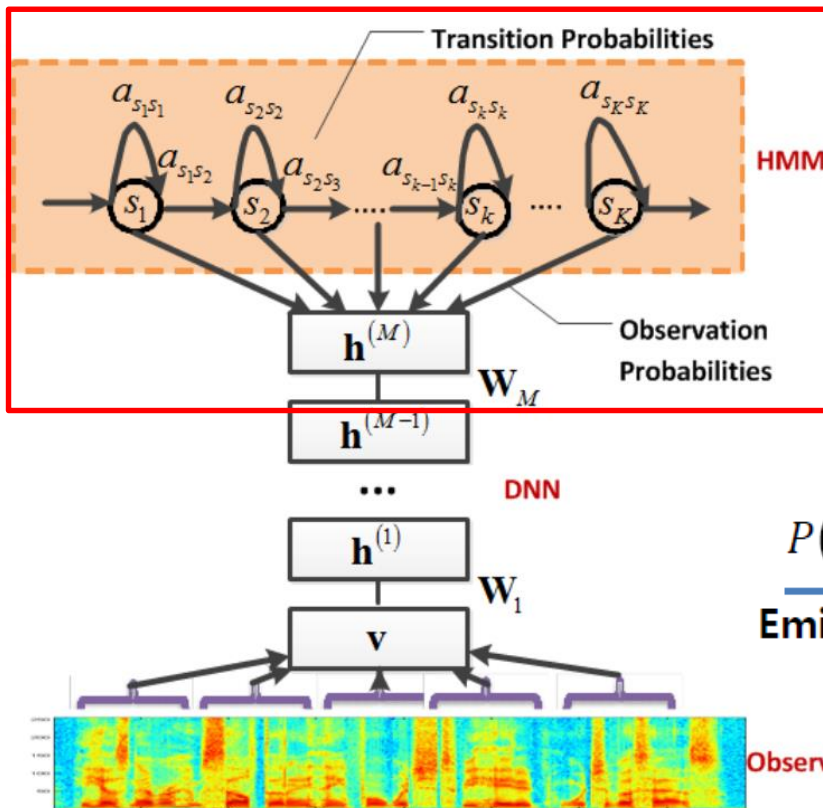
## 5. 한국어 음성인식 구현에 있어서의 이슈

- MFCC 추출과정에서의 언어에 따른 이슈: 없음



# 5. 한국어 음성인식 구현에 있어서의 이슈

- 음향모델에서의 언어에 따른 이슈
  - ◆ 음소의 정의외에는 없음



**Network output**

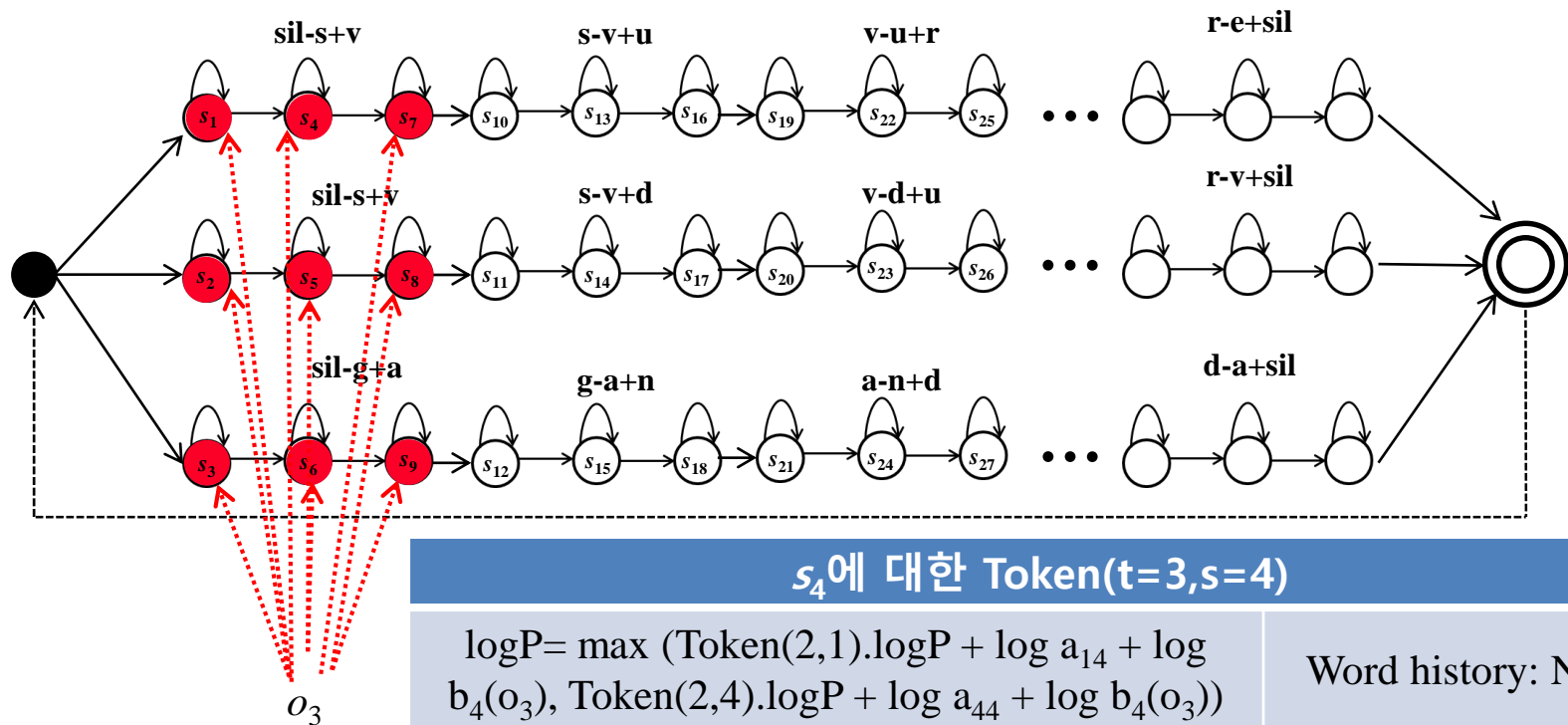
$$P(o_i | s^k) = \frac{P(s^k | o_i) P(o_i)}{P(s^k)} \approx \frac{P(s^k | o_i)}{P(s^k)}$$

**Emission prob.**

[Dahl, 2012]

# 5. 한국어 음성인식 구현에 있어서의 이슈

## ■ 디코딩에서의 이슈: 없음



## 5. 한국어 음성인식 구현에 있어서의 이슈

- 언어모델 및 어휘에 있어서의 언어에 따른 이슈
  - ◆ 단어
  - ◆ 단어의 경계 (어절/형태소)
  - ◆ 형태소 분석기 필요 여부

$$P(W) = \prod_{k=1}^T P(w_k | w_{k-1} w_{k-2} \dots w_0)$$

- 발음열 생성에서의 언어에 따른 이슈: 한국어 발음법에 적용되는 규칙이 생각보다 많음



## 5.1 발음열 변환 (Grapheme to Phoneme (G2P))

- 한국어 발음법은 '2014년 개정 한국어 표준발음법 규칙'에 잘 정리되어 있음
  - ◆ 크게 20개의 규칙으로 정리할 수 있음

종류	내용
일반 규칙(총 8개)	음절말 중화, 자음군 단순화, 격음화, 유음화, 장애음의 비음화, 유음의 비음화, 경음화, 자음의 첫소리 'ㄴ' 모음화
형태소 분석이 필요한 규칙(총 5개)	연음법칙, 구개음화, 종성 ㅎ-탈락, ㄴ-첨가, 용언의 활용형 'ㄷ' 모음화
수의적 발음규칙(총 7개)	동일 조음위치 자음 탈락, 중복 자음화, 변자음화, 초성 ㅎ-탈락, 'ㄷ'모음화, 단어 첫음절 외 '의' 모음화, 용언 어미 '어' 모음화

## 5.1 발음열 변환 (Grapheme to Phoneme (G2P))

### ■ 한글 외의 문자에 대한 처리

#### ◆ 숫자

- 단위명사 종류에 따라 기수로 읽을수 있고, 서수로 읽을수 있음  
예: '123명의 사람' => '백이십삼명의 사람'  
예: '1개' => '한 개', '1마리'=> '한마리'

#### ◆ 영문자

#### ◆ 한글 외의 문자에 대한 처리

- 마침표(.), 쉼표(,), 물음표(?), 느낌표(!)
- 그 외 기호, 문자

## 5.1 발음열 변환 (Grapheme to Phoneme (G2P))

◆ 예:

- 이번에 새로나오는거 어떤가 보고있다 난 폰카 이쁘게 찍히는게 넘나 부러움
  - 수의적 발음 규칙 미적용
    - » 이번에 새로나오는거 어떤가 보고있다 난 폰카 이쁘게 찍히는게 넘나 부러움
  - 수의적 발음 규칙 적용
    - » 이번에 새로나오는거 어떻게 보고있다 난 폰카 입쁘게 찍기능게 넘나부러움
- 수의적 발음 규칙은 적용할 수도 있고 적용하지 않을수도 있는 규칙임
  - » 빠른 발음 주로 적용됨

## 5.1 발음열 변환 (Grapheme to Phoneme (G2P))

### ◆ 예문에 적용된 규칙:

- 이번에 새로나오는거 어떻게 보고있다 난 폰카 입쁘게 찍기능게 넘나부러움
  - 새로나오는거 => 새로나오는거
    - » 수의적 발음 규칙 중 '변자음화' 적용
  - 어떨가 => 어떻게
    - » 일반 규칙 중 '음절말 중화' 적용
    - » 수의적 발음 규칙 중 '변자음화' 적용
  - 보고있다 => 보고있다
    - » 일반 규칙 중 '음절말 중화' 적용
  - 폰카 => 폰카
    - » 수의적 발음 규칙 중 '변자음화' 적용
  - 이쁘게 => 입쁘게
    - » 수의적 발음 규칙 중 '중복자음화' 적용
    - » L3: #, R1:ㅁㅁ => L3: ㅍ, R1: ㅁㅁ

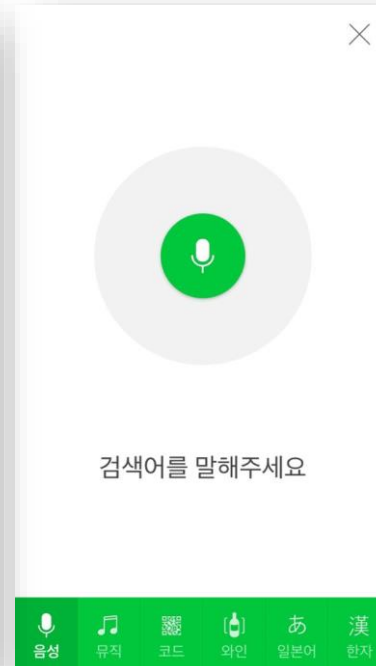
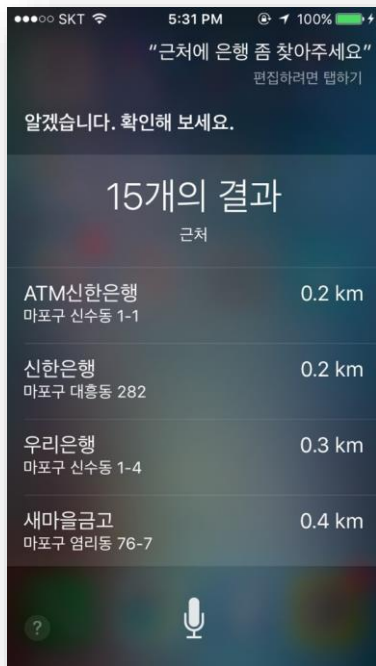
## 5.1 발음열 변환 (Grapheme to Phoneme (G2P))

### ◆ 예문에 적용된 규칙

- 이번에 새로나오능거 어떻게 보고있다 난 풍카 입쁘게 짹키능게 넘 나부러움
  - 짹히는게 => 짹키능게
    - » 일반 규칙 중 '격음화' 적용
    - » 수의적 발음 규칙 중 '변자음화' 적용

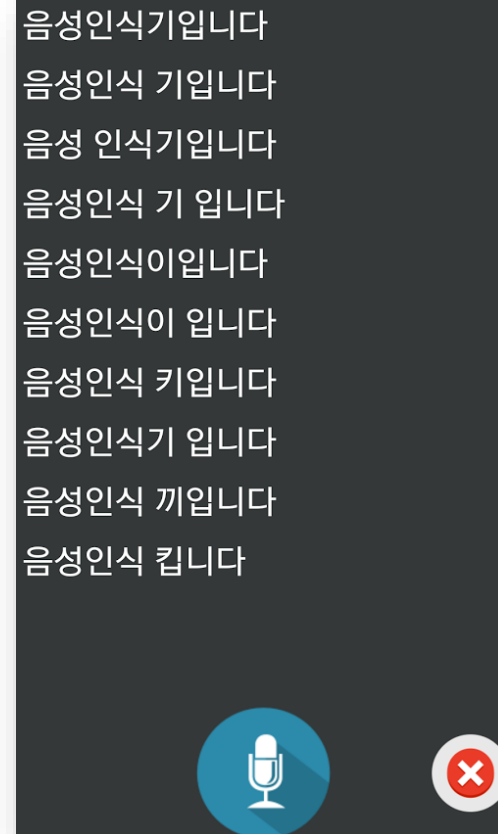
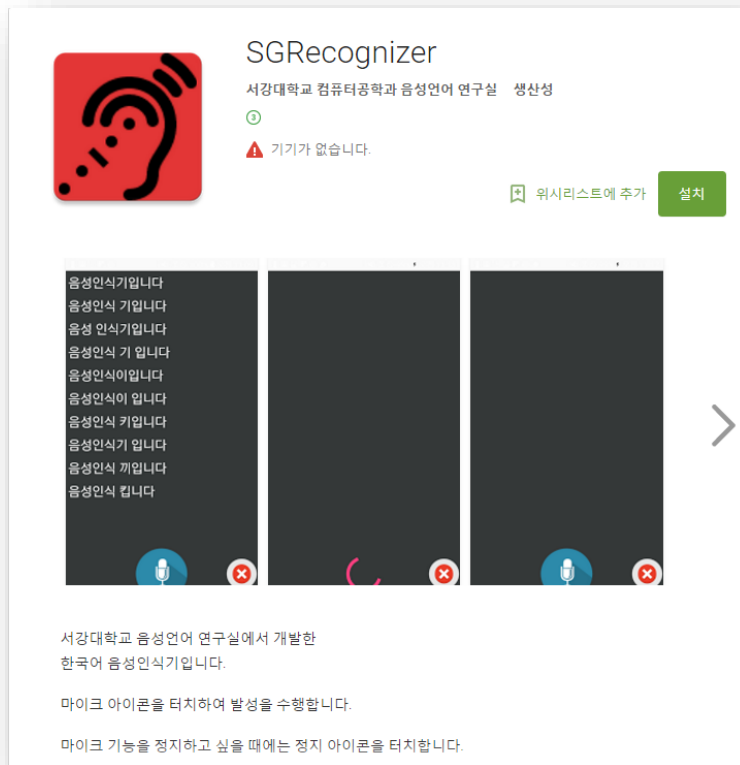
# 6.1 휴대폰용 음성인식 앱

- SIRI, 네이버 음성검색, S voice, Q voice 등



# 6.1 휴대폰용 음성인식 앱

- 서강대학교 한국어 음성인식기 'SGRecognizer'
- ◆ 다운로드 : 구글 플레이스토어 'SGRecognizer' 검색



# 6.2 음성인식기 I/O 라이브러리

## ■ Google speech API

- ◆ Google에서 제공하는 STT(Speech-to-Text) android API
- ◆ Android.speech.RecognitionListener를 import하여 사용함
  - <http://developer.android.com/intl/ko/reference/android/speech/package-summary.html>

The screenshot shows the Android Developers website for the `android.speech` package. The page is titled "package android.speech" and indicates it was "Added in API level 3". The left sidebar contains a navigation menu with sections for "Android APIs", "Interfaces", and "Classes". The main content area lists the "Interfaces" and "Classes" available in the package.

**Interfaces**

Interface	Description
<a href="#">RecognitionListener</a>	Used for receiving notifications from the SpeechRecognizer when the recognition related events occur.

**Classes**

Class	Description
<a href="#">RecognitionService</a>	This class provides a base class for recognition service implementations.
<a href="#">RecognitionService.Callback</a>	This class receives callbacks from the speech recognition service and forwards them to the user.
<a href="#">RecognizerIntent</a>	Constants for supporting speech recognition through starting an <a href="#">Intent</a> .
<a href="#">RecognizerResultsIntent</a>	Constants for intents related to showing speech recognition results.
<a href="#">SpeechRecognizer</a>	This class provides access to the speech recognition service.

At the bottom of the page, there is a "Get news & tips" section with a mail icon, and social media links for "Blog" and "Support" with YouTube, Google+, and Twitter icons. The footer contains legal information and a language selector set to "한국어".



## 6.2 음성인식기 I/O 라이브러리

### ■ Amazon Alexa Voice Service API

- ◆ 개발자가 Amazon Echo를 이용하여 customized service를 지원하도록 하는 음성인식 API
- ◆ API LINK: <https://developer.amazon.com/public/solutions/alexa/alexa-voice-service/content/avs-api-overview>

The screenshot shows the Amazon developer portal for the Alexa Voice Service API. The top navigation bar includes the Amazon logo, a search bar, and links for 'SIGN IN or CREATE FREE ACCOUNT' and 'ENGLISH'. Below this is a secondary navigation bar with links for 'HOME', 'ALEXA', 'SERVICES & APIS', 'DEVICES', 'RESOURCES', 'BLOG', and 'SUPPORT'. The 'ALEXA' link is highlighted. The main content area is titled 'Alexa Voice Service API Overview (v20160207)'. On the left, there is a sidebar with links for 'Alexa', 'Alexa Skills Kit', 'Alexa Voice Service' (highlighted), and 'Alexa Fund'. The main content area contains a table of contents with links for 'Overview', 'Authorization', 'Transport Protocol', 'Base URL', 'Interfaces', 'Versioning', and 'Need Help?'. Below this is the 'Overview' section, which states: 'The Alexa Voice Service (AVS) API allows developers to voice-enable connected products with a microphone and speaker. Once integrated, your product will have access to the built in capabilities of Alexa (like music playback, timers and alarms, package tracking, movie listings, calendar management, and more) and third-party skills developed using the Alexa Skills Kit.' On the right side, there is a section titled 'AVS DOCUMENTATION' with a sub-section 'API + Reference' containing links for 'Overview', 'SpeechRecognizer Interface', 'SpeechSynthesizer Interface', 'Alerts Interface', 'AudioPlayer Interface', 'PlaybackController Interface', 'Speaker Interface', 'System Interface', and 'Context'.

## 6.2 음성인식기 I/O 라이브러리

### ■ Daum Newton API

- ◆ TTS(Text-to-Speech), STT(Speech-to-Text) 제공
- ◆ API LINK: <http://developers.daum.net/services/apis/newtone/>

## 6.2 음성인식기 I/O 라이브러리

### ■ Naver 음성인식 API

- ◆ 한국어 및 영어에 대한 TTS 제공
- ◆ API LINK: <https://developers.naver.com/docs/labs/vrecog>

#### 음성인식 API 명세

사람의 목소리를 소켓 통신을 이용해 스트리밍 형태로 전달하면 서버에서 인식해 텍스트로 리턴해주는 API입니다. HTTP 기반의 REST API가 아니며 자체 스트리밍 프로토콜을 구현한 SDK가 제공됩니다. 현재는 Android 버전의 SDK만 제공되며, iOS용 SDK는 조만간 제공될 예정입니다.

REST API는 대부분 서버에서 Client ID와 Client Secret을 이용하여 인증하는데요, 음성인식 SDK를 이용한 App은 외부에 배포되는 노출의 위험이 있기 때문에, 인증 시 Client ID와 Android App의 개발 패키지 이름을 이용합니다.

[오픈 API 이용 신청 >](#)[Android SDK 및 샘플코드 >](#)[Android API Document](#)[iOS SDK 및 샘플코드 >](#)

#### API 호출 예제

예제 실행 전에 아래 **1.준비사항** 항목들을 꼭 체크하시길 바랍니다.

Android

iOS


```
// 네이버 음성인식 Open API 예제 - Android
// 풀소스코드 링크 https://github.com/naver/naverspeech-sdk-android
// 안드로이드는 2가지 핵심 클래스로 구성되어 있습니다. 자세한건 github 예제코드를 참조하세요.
// 1. MainActivity 클래스 - SpeechRecognitionListener를 초기화하고, 이후 이벤트를 handleMessage 에서 받아 처리 (예: https://github.com/naver/naverspeech-s)
// 2. SpeechRecognitionListener 를 상속한 클래스 - 음성인식 서버 연결, 음성전달, 인식결과 발생등의 이벤트에 따른 결과 처리 방법 정의 (예: https://github.com/i)
// 1. MainActivity 클래스
public class MainActivity extends Activity {
    private static final String TAG = MainActivity.class.getSimpleName();
    private static final String CLIENT_ID = "YOUR CLIENT ID"; // "내 애플리케이션"에서 Client ID를 확인해서 이곳에 적어주세요.
    private RecognitionHandler handler;
    private NaverRecognizer naverRecognizer;
    private TextView txtResult;
    private Button btnStart;
    private String mResult;
```

## 6.2 음성인식기 I/O 라이브러리

### ■ 솔트룩스 아담 인텔리전스 API

- ◆ 데이터, 분석, 언어, 지식, 지능으로 구성된 60여개의 API를 제공
- ◆ 현재 closed beta 형태로 파트너 협약을 맺은 기업/기관을 대상으로만 서비스를 공개하고 있음

- ◆ API LINK: <http://adams.ai/>



〈 챗봇과 질의응답 시스템 〉

〈 가상현실과 지식검색 〉

〈 IoT와 임베디드 서비스 〉


〈 퀴즈와 엔터테인먼트 〉

아담 인텔리전스 서비스 사용하기

## 6.3.1 음성인식기 개발 툴

### ■ HTK (Hidden Markov Toolkit) [Young, 2002]

◆ 출처 웹 주소: [htk.eng.cam.ac.uk](http://htk.eng.cam.ac.uk)



Home | Register | Mailing Lists | Documentation

Home

**Getting HTK**  
[Register](#)  
[Manage login/password](#)  
[Download](#)

**Documentation**  
[HTKBook](#)  
[FAQ](#)  
[History of HTK](#)  
[CUED LVR Systems](#)  
[License](#)

**Mailing Lists**  
[Subscribe](#)  
[Account/Unsubscribe](#)  
[Archives](#)

**Development**  
[Get involved](#)  
[Future Plans](#)  
[Report a Bug](#)  
[Bug Status](#)  
[ATK](#)

### What is HTK?

The Hidden Markov Model Toolkit (HTK) is a portable toolkit for building and manipulating hidden Markov models. HTK is primarily used for speech recognition research although it has been used for numerous other applications including research into speech synthesis, character recognition and DNA sequencing. HTK is in use at hundreds of sites worldwide.

HTK consists of a set of library modules and tools available in C source form. The tools provide sophisticated facilities for speech analysis, HMM training, testing and results analysis. The software supports HMMs using both continuous density mixture Gaussians and discrete distributions and can be used to build complex HMM systems. The HTK release contains extensive documentation and examples.

HTK was originally developed at the [Machine Intelligence Laboratory](#) (formerly known as the Speech Vision and Robotics Group) of the Cambridge University Engineering Department (CUED) where it has been used to build CUED's large vocabulary speech recognition systems (see [CUED HTK LVR](#)). In 1993 Entropic Research Laboratory Inc. acquired the rights to sell HTK and the development of HTK was fully transferred to Entropic in 1995 when the Entropic Cambridge Research Laboratory Ltd was established. HTK was sold by Entropic until 1999 when Microsoft bought Entropic. Microsoft has now licensed HTK back to CUED and is providing support so that CUED can redistribute HTK and provide development support via the HTK3 web site. See [History of HTK](#) for more details.

While Microsoft retains the copyright to the original HTK code, everybody is encouraged to make changes to the source code and contribute them for inclusion in HTK3.




# 6.3.1 음성인식기 개발 툴






## ■ Sphinx4 [Lamere, 2003]

### ◆ 출처 웹 주소:

<http://cmusphinx.sourceforge.net/wiki/tutorialspinx4>



Open Source Toolkit For Speech Recognition  
Project by [Carnegie Mellon University](#)

DownloadLearnResearchDevelopCommunicate

Show pagesourceOld revisionsRecent changesSearch

Trace: > [Sphinx-4 Application Programmer's Guide](#)

### Sphinx-4 Application Programmer's Guide

WARNING: THIS TUTORIAL DESCRIBES SPHINX4 API FROM THE PRE-ALPHA RELEASE:  
<https://sourceforge.net/projects/cmusphinx/files/sphinx4/5%20prealpha/>

The API described here is not supported in earlier versions

#### Overview

Sphinx-4 is a pure Java speech recognition library. It's very flexible in its configuration, and in order to carry out speech recognition jobs quite a lot of objects depending on each other should be instantiated, throughout this article we will call them all together "object graph". Fortunately, the most of the objects can be instantiated automatically, and for those few requiring manual setup Sphinx-4 provides high-level interfaces and a context class that takes out the need to setup each parameter of the object graph separately.

#### Using in your projects

Sphinx-4 is available as a maven package in [Sonatype OSS repository](#). To use sphinx4 in your maven project specify this repository in your pom.xml:

```
<project>
...
<repositories>
  <repository>
```

#### Table of Contents

- [Sphinx-4 Application Programmer's Guide](#)
  - [Overview](#)
  - [Using in your projects](#)
  - [Basic Usage](#)
    - [Configuration](#)
    - [LiveSpeechRecognizer](#)
    - [StreamSpeechRecognizer](#)
    - [SpeechAligner](#)
    - [SpeechResult](#)
  - [API Extension](#)
    - [Context](#)
    - [AbstractSpeechRecognizer](#)
  - [XML Configuration](#)
  - [Additional Information](#)

## 6.3.1 음성인식기 개발 툴

### ■ Kaldi [Povey, 2011]

- ◆ DNN-HMM 방식 지원
- ◆ C++ 기반의 객체 지향 형태로 구현되어 있어, 새로운 입력 및 내부 구조 변경에 대해 확장 가능성 있도록 구현함
- ◆ 디코딩 과정에서 WFST 활용
  - Code-level integration with Finite State Transducers (FSTs)
- ◆ LDC에서 제공하는 음성 코퍼스를 이용한 Recipe를 제공
  - 대표적 영어 연속 음성인식 Recipe : WSJ, RM
- ◆ 새로운 SW 개발시 소스코드 공개 의무 없으며, 상업적 이용에 제한이 없음

# 6.3.1 음성인식기 개발 툴

## ■ Kaldi

◆ 출처 웹 주소: <http://kaldi.sourceforge.net/about.html>

### 'KALDI'

The screenshot displays the Kaldi project website. At the top, there is a navigation bar with tabs: 'Main Page', 'Related Pages', 'Modules', 'Namespaces', 'Classes', and 'Files'. A search bar is located on the right side of this bar. Below the navigation bar, the left sidebar contains a tree view of the project structure, with 'Kaldi' selected. The main content area, titled 'Kaldi', contains a paragraph of text and a list of links. The text states: 'Please see the [instructions](#) on upgrading your repository to the new location, following our upgrade to the "new" Sourceforge, (see also Kaldi's project page on Sourceforge, and [kaldi-asr.org](#) where you can download pre-built models,'). The list of links includes: 'About the Kaldi project', 'Other Kaldi-related resources', 'Downloading and installing Kaldi', 'Software required to install and run Kaldi', 'Legal stuff', 'Plans for Kaldi development', 'Kaldi tutorial', 'Data preparation', 'The build process (how Kaldi is compiled)', 'The Kaldi coding style', 'Contacting the Kaldi team', 'History of the Kaldi project', 'The Kaldi Matrix library', 'External matrix libraries', 'The CUDA Matrix library', 'Kaldi I/O mechanisms', 'Kaldi I/O from a command-line perspective', 'Kaldi logging and error-reporting', 'Parsing command-line options', 'Other Kaldi utilities', 'Clustering mechanisms in Kaldi', and 'HMM topology and transition modeling'. At the bottom right of the page, there is a footer that reads: 'Generated on Fri Oct 31 2014 21:06:55 for 'KALDI' by [doxygen](#) 1.8.1.2'.

Main Page Related Pages Modules Namespaces Classes Files Search

▼ 'KALDI'

Kaldi

▶ About the Kaldi project  
Other Kaldi-related resources  
▶ Downloading and installing Kaldi  
▶ Software required to install and run Kaldi  
Legal stuff  
▶ Plans for Kaldi development  
▶ Kaldi tutorial  
▶ Data preparation  
▶ The build process (how Kaldi is compiled)  
The Kaldi coding style  
Contacting the Kaldi team  
▶ History of the Kaldi project  
▶ The Kaldi Matrix library  
▶ External matrix libraries  
▶ The CUDA Matrix library  
▶ Kaldi I/O mechanisms  
▶ Kaldi I/O from a command-line perspective  
▶ Kaldi logging and error-reporting  
▶ Parsing command-line options  
▶ Other Kaldi utilities  
▶ Clustering mechanisms in Kaldi  
▶ HMM topology and transition modeling

Kaldi

Please see the [instructions](#) on upgrading your repository to the new location, following our upgrade to the "new" Sourceforge, (see also Kaldi's project page on Sourceforge, and [kaldi-asr.org](#) where you can download pre-built models,).

- [About the Kaldi project](#)
- [Other Kaldi-related resources](#)
- [Downloading and installing Kaldi](#)
- [Software required to install and run Kaldi](#)
- [Legal stuff](#)
- [Plans for Kaldi development](#)
- [Kaldi tutorial](#)
- [Data preparation](#)
- [The build process \(how Kaldi is compiled\)](#)
- [The Kaldi coding style](#)
- [Contacting the Kaldi team](#)
- [History of the Kaldi project](#)
- [The Kaldi Matrix library](#)
- [External matrix libraries](#)
- [The CUDA Matrix library](#)
- [Kaldi I/O mechanisms](#)
- [Kaldi I/O from a command-line perspective](#)
- [Kaldi logging and error-reporting](#)
- [Parsing command-line options](#)
- [Other Kaldi utilities](#)
- [Clustering mechanisms in Kaldi](#)
- [HMM topology and transition modeling](#)

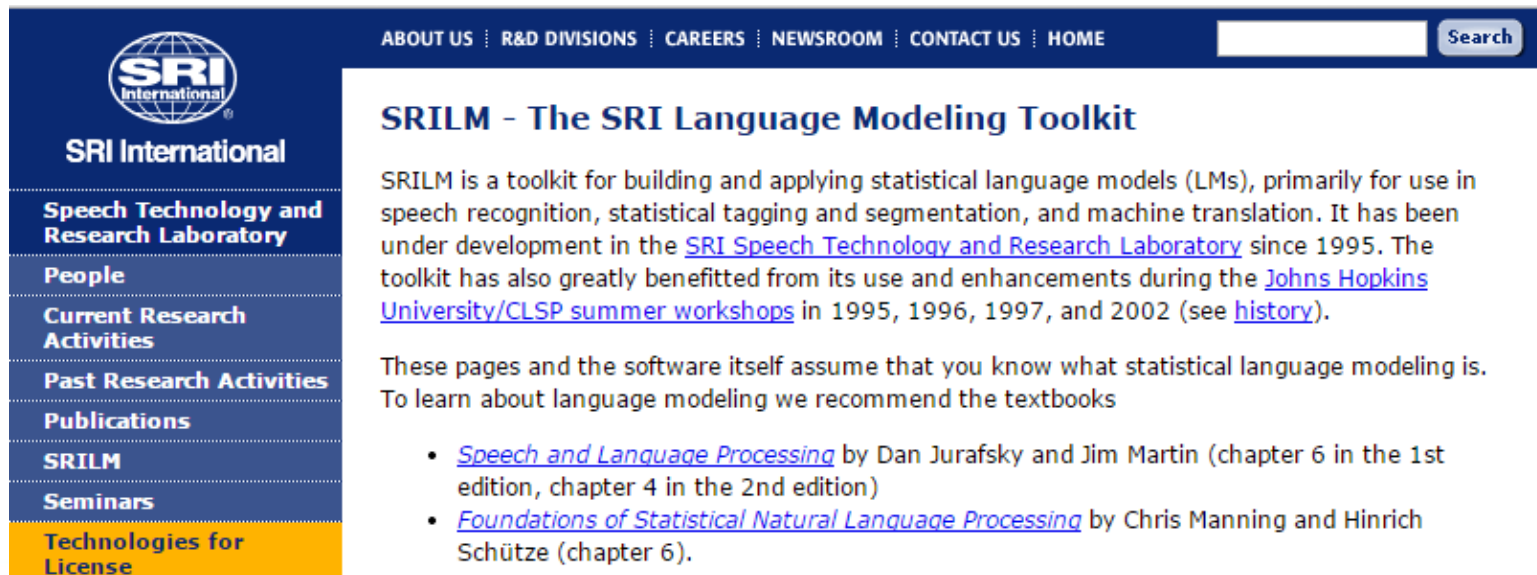
Generated on Fri Oct 31 2014 21:06:55 for 'KALDI' by [doxygen](#) 1.8.1.2



## 6.3.2 언어모델 개발 툴

### ■ SRILM [Stolcke, 2002]

- ◆ 출처 웹 주소: <http://www.speech.sri.com/projects/srilm/>
- ◆ 활용 범위: 음성인식 및 기계 번역, 등
- ◆ 단어 기반의 back-off 언어모델 생성 (entropy기반의 pruning 제공)
- ◆ Class-based 언어모델 생성 가능 및 dynamic 또는 static interpolation 방법 제시



The screenshot shows the SRILM website. The header includes the SRI International logo and a navigation bar with links: ABOUT US, R&D DIVISIONS, CAREERS, NEWSROOM, CONTACT US, HOME. A search bar is also present. The left sidebar lists various categories: SRI International, Speech Technology and Research Laboratory, People, Current Research Activities, Past Research Activities, Publications, SRILM, Seminars, and Technologies for License. The main content area is titled 'SRILM - The SRI Language Modeling Toolkit' and contains a paragraph describing the toolkit's purpose and history, followed by a list of recommended textbooks.

**SRILM - The SRI Language Modeling Toolkit**

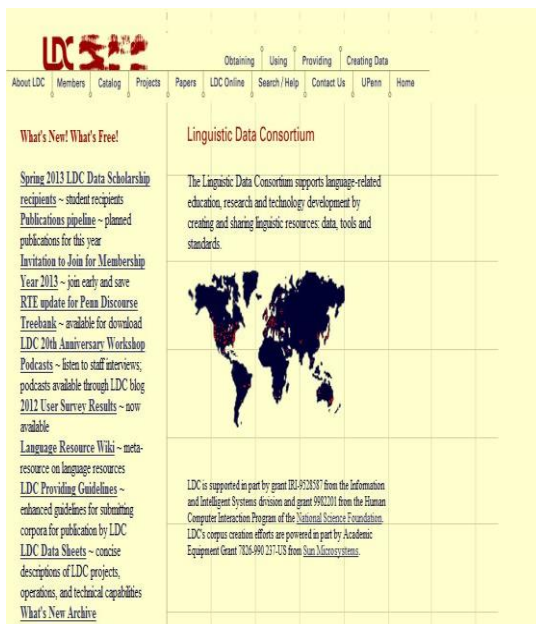
SRILM is a toolkit for building and applying statistical language models (LMs), primarily for use in speech recognition, statistical tagging and segmentation, and machine translation. It has been under development in the [SRI Speech Technology and Research Laboratory](#) since 1995. The toolkit has also greatly benefitted from its use and enhancements during the [Johns Hopkins University/CLSP summer workshops](#) in 1995, 1996, 1997, and 2002 (see [history](#)).

These pages and the software itself assume that you know what statistical language modeling is. To learn about language modeling we recommend the textbooks

- [Speech and Language Processing](#) by Dan Jurafsky and Jim Martin (chapter 6 in the 1st edition, chapter 4 in the 2nd edition)
- [Foundations of Statistical Natural Language Processing](#) by Chris Manning and Hinrich Schütze (chapter 6).

## 6.3.3 음성 코퍼스

- 영어 음성 corpus 관련 사이트
  - ◆ LDC (Linguistic Data Consortium. [www ldc.upenn.edu](http://www ldc.upenn.edu))
  - ◆ SpeechOcean ([www.speechocean.com](http://www.speechocean.com))



LDC



SpeechOcean

## 6.3.3 음성 코퍼스

- 한국어 음성 corpus 관련 사이트
  - ◆ 원광대학교 SiTEC ([www.sitec.or.kr](http://www.sitec.or.kr))
  - ◆ ETRI ([https://itec.etri.re.kr/itec/sub02/sub02\\_01.do](https://itec.etri.re.kr/itec/sub02/sub02_01.do))

**SiTEC** 원광대학교  
음성정보기술산업지원센터  
Speech Information Technology & Industry Promotion Center  
<http://www.sitec.or.kr/>

Home About 음성 DB 리스트 Gallery Contact

배포 중인 음성 DB 리스트입니다.

< Speech resources which are being distributed at SiTEC >

Types	Titles	Contents	Speakers
		<ul style="list-style-type: none"><li>• speech recorded simultaneously through 7 microphones and 1 hands-free in car driving environments</li><li>• car at 2,000CC, 2,500CC, RV</li><li>• city driving(40~60km/h), highway driving(70~90km/h)</li><li>• single digits, audio control commands, command words for telematics /navigation, address, POA commands, place names, train and subway stations</li></ul>	1,600
		<ul style="list-style-type: none"><li>• recording sessions time gap: 1st recording, 1 day after</li></ul>	

원광대학교 SiTEC

**ETRI** 기술이전홈페이지  
LOGIN JOIN SITEMAP ETRI

기술이전안내 기술이전검색/신청 기술이전설명회 기술 예고 자료실 고객지원

고객과 함께 꿈을 실현하는 ETRI

Technology Transfer  
기술이전검색/신청

기술이전검색  
기술이전관리신청

기술이전검색/신청은 쉽고 빠르게 검색하실 수 있습니다.  
기술과 관련된 검색어를 입력하시고 검색 버튼을 클릭하세요.

기간별 → ~  
연구분야별 → 전체  
기술분야별 → 전체  
일반검색 → 기술명

[TOTAL : 1627] [PAGE : 1/163]

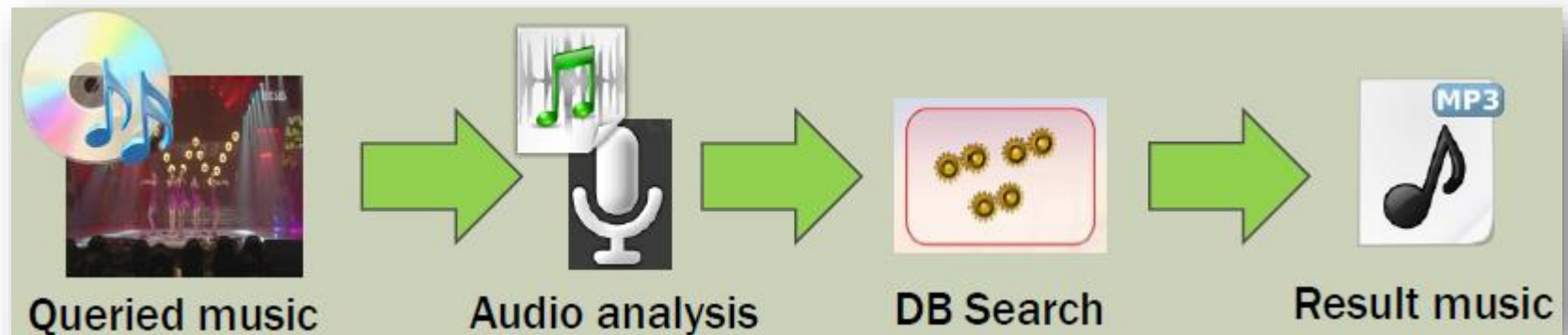
번호	기술명	첨부	작성자	날짜
1627	서버형 여행용 한/영 자동통역 기술 2014 - 시험용	첨	염상원	2014-11-26
1626	서버형 여행용 한/중 자동통역 기술 2014 - 시험용	첨	염상원	2014-11-26
1625	서버형 여행용 한/일 자동통역 기술 2014 - 시험용	첨	염상원	2014-11-26

ETRI

## 7.1.1 음악인식 소개

### ■ 오디오 정보를 이용한 음악 인식

- ◆ 오디오 신호 분석을 통해 기존 메타데이터(제목, 가수) 정보 없이 사용자가 질의한 음악에 대한 결과를 제공
- ◆ 제목, 가수를 몰라도 음악을 녹음하여 검색
- ◆ 스마트폰 대상의 서비스 구현시 사용자 접근성, 편의성 증가



# 7.1.1 음악인식 소개

## ■ 음악 인식의 활용 분야

- ◆ 방송 신호 모니터링
- ◆ 방송 관련 정보 연계 제공
- ◆ 파일 공유 시스템의 불법자료 필터링 시스템
- ◆ 자동화된 음악 정보 시스템 조직
- ◆ 음악 검색 서비스

## 7.1.1.1 음악 검색 서비스 소개

### ■ 음악 검색 서비스

- ◆ 사용자가 녹음한 오디오 신호를 분석
- ◆ 음악서비스 DB에서 검색
- ◆ 검색 결과와 함께 관련 자료(VOD, 음악구매)를 제공
  - 장점: 음악을 쉽고 빠르게 검색 가능하고 관련 자료 제공 시 구매로 이어질 가능성이 높음

### ■ 음악 검색 서비스

- ◆ 국내
  - 네이버 음악검색 (<http://mobile.naver.com/application/naver.nhn>)
- ◆ 해외
  - Gracenote (<http://www.gracenote.com>)
  - Shazam(<http://www.shazam.com>)
  - Midomi (<http://www.midomi.com>)



## 7.1.1.2 음악인식 알고리즘

- 상용 음악검색 서비스 중 가장 널리 사용되는 서비스의 구현방법
  - ◆ Shazam: audio fingerprint 기반 (Shazam encore) [1, 2, 3, 4, 9]
    - 구현 방법: audio fingerprint 기반 [1, 2, 3, 4, 9]
    - 음악검색 서비스 이름: Shazam encore
  - ◆ Gracenote: audio fingerprint 기반 (Mobile MusicID) [5, 6, 9]
    - 구현 방법: audio fingerprint 기반 [5, 6, 9]
    - 음악검색 서비스 이름: Mobile MusicID
  - ◆ Midomi: query-by-humming 기반 (Soundhound) [8, 9]
    - 구현 방법: query-by-humming 기반 [8, 9]
    - 음악검색 서비스 이름: Soundhound

## 7.1.2 Audio fingerprint 음악인식 알고리즘

### ■ Gracenote (2.1장에서 소개)

- ◆ J. Haitsma and T. Kalker, "A Highly Robust Audio Fingerprinting System," In *Proc. International Symposium on Music Information Retrieval (ISMIR)*, Oct. 2002, pp.144-147.
- ◆ J. Haitsma, A. Kalker, and S. Schimmel, "Efficient Storage of Fingerprints," U.S. Patent 7477739.

### ■ Shazam (2.2장에서 소개)

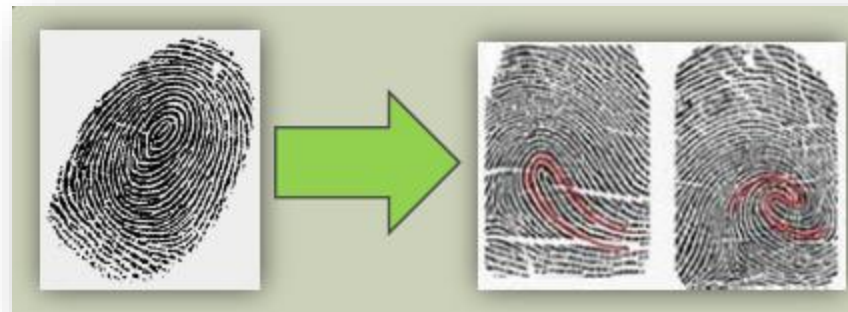
- ◆ A. Wang, "An Industrial Strength Audio Search Algorithm," In *Proc. International Symposium on Music Information Retrieval (ISMIR)*, Oct. 2003, pp.7-13.
- ◆ A. Wang, "An Industrial Strength Audio Search Algorithm," In *Proc. International Symposium on Music Information Retrieval (ISMIR)*, Oct. 2003, pp.7-13.



## 7.1.2 Audio fingerprint 음악인식 알고리즘

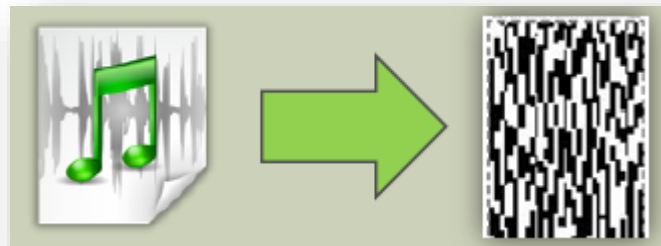
### ■ Fingerprint

- ◆ 지문: 사람의 손가락 마디에 있는 고유한 무늬로써 신분 확인에 사용



### ■ Audio fingerprint

- ◆ 오디오 신호가 가지는 고유한 신호 패턴의 특징을 추출할 수 있다면 인지적으로 동일하게 느껴지는 오디오 신호에 대한 검색이 가능함



## 7.1.2 Audio fingerprint 음악인식 알고리즘

### ■ Audio fingerprint 방식의 장점

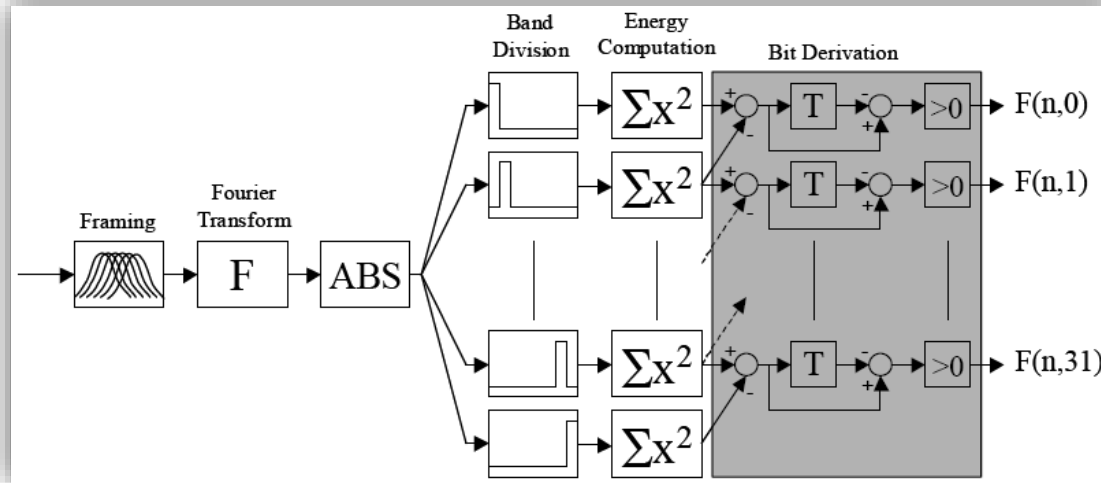
- ◆ 오디오 신호에 비해 작은 크기로 저장되어 오디오 신호 자체를 이용하는 것 보다 메모리와 저장공간의 절약이 가능
- ◆ 데이터 집합이 작아짐으로써 효율적인 검색 가능

# 7.1.2.1 Gracenote 구현방법

## ■ 오디오 fingerprint

### ◆ 오디오 신호로부터 밴드에너지 특징 추출

- 오디오 신호로부터 0.37초의 길이를 가지는 오버랩된 프레임들로 자름
- 11.6ms 마다 해밍 윈도우를 씌우고 Fourier transform을 함
- 결과 값을 절대 값으로 변환
- 33개의 겹치지 않는 주파수 밴드 값으로 나뉘며 밴드별 에너지를 계산



Overview of fingerprint extraction scheme ([5])

## 7.1.2.1 Gracenote 구현방법

### ■ 밴드에너지의 오디오 fingerprint 변환

- ◆ 프레임 단위: 11.6ms (프레임마다 하나의 sub-fingerprint 추출)
- ◆ 프레임 단위로 FFT를 거쳐 33개의 bin에서 각 bin의 에너지 계산
- ◆ sub-fingerprint는 에너지 밴드열 에너지 값을 바탕으로 생성된 32개의 0/1값
- ◆ 256개의 연속된 sub-fingerprint가 하나의 fingerprint block을 형성(약 3 초)
- ◆ 300Hz~2000Hz범위의 logarithmic 공간을 가지는 주파수 대역을 선정
- ◆  $E(n,m)$ : n번째 프레임의 m번째 밴드 에너지
- ◆  $F(n,m)$ : n번째 프레임의 m번째 bit의 sub-fingerprint값
- ◆ 아래 식에 의해 sub-fingerprint의 비트들이 정해짐

$$F(n,m) = \begin{cases} 1 & \text{if } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) > 0 \\ 0 & \text{if } E(n,m) - E(n,m+1) - (E(n-1,m) - E(n-1,m+1)) \leq 0 \end{cases}$$

**Fig 9. Formula of F(n, m) function**

## 7.1.2.1 Gracenote 구현방법

### ■ 오디오 fingerprint 특성

- ◆ 원음의 오디오 신호와 압축된 MP3 파일의 비트 에러율이 크지 않음
  - 128Kbps MP3 파일의 경우  $BER = 0.078$

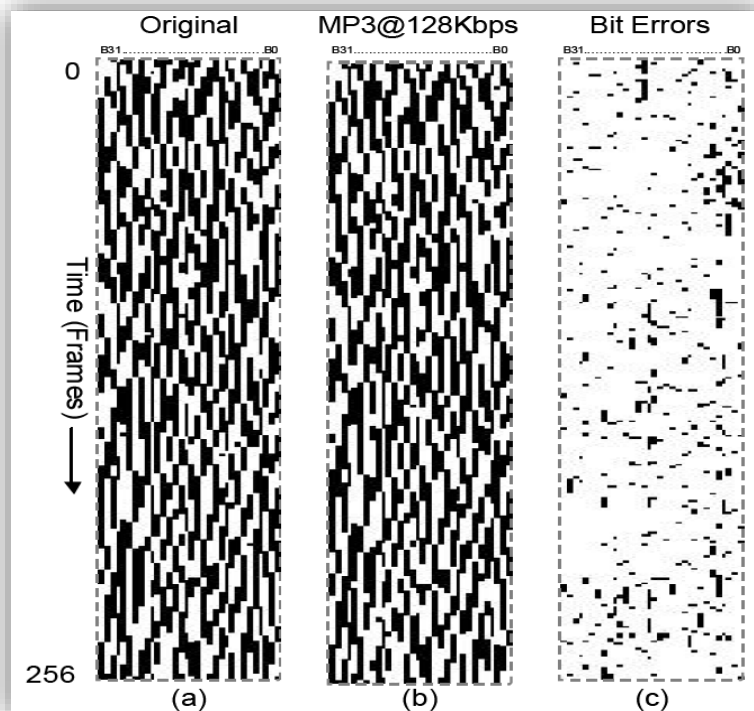


Fig 11. (a) Fingerprint block of original music clip,  
(b) fingerprint block of a compressed version,  
(c) The difference between a and b showing the bit errors in black (BER=0.078). ([5])

## 7.1.2.1 Gracenote 구현방법

### ■ 음악검색 DB 설계

- ◆ 음악들은 fingerprint로 변환되어 DB에 저장되고 Fig 12.와 같은 형태로 저장
- ◆ Look-Up Table(LUT)는 해당 sub-fingerprint가 음악 DB에서 어디에 저장되어 있는지에 대한 정보를 포함
- ◆ 제한된 메모리에서  $2^{32}$ 개의 입력을 포함하는 LUT는 램에 상주하며 데이터가 sparse한 형태로 저장되기 때문에, hash function을 사용해 LUT를 구현

### ■ 음악검색 방법

- ◆ 입력된 음악을 fingerprint로 변환하여 fingerprint는 LUT의 인덱스로써 참조
- ◆ 모든 fingerprint에 해당하는 인덱스를 참조하여, 인덱스에 연결된 노드들의 곡 정보와 시간 정보를 저장
- ◆ 탐색 노드 중에서 가장 많이 발견되고, fingerprint의 sequence를 가지고 있는 음악이 검색 결과로 제공

## 7.1.2.1 Gracenote 구현방법

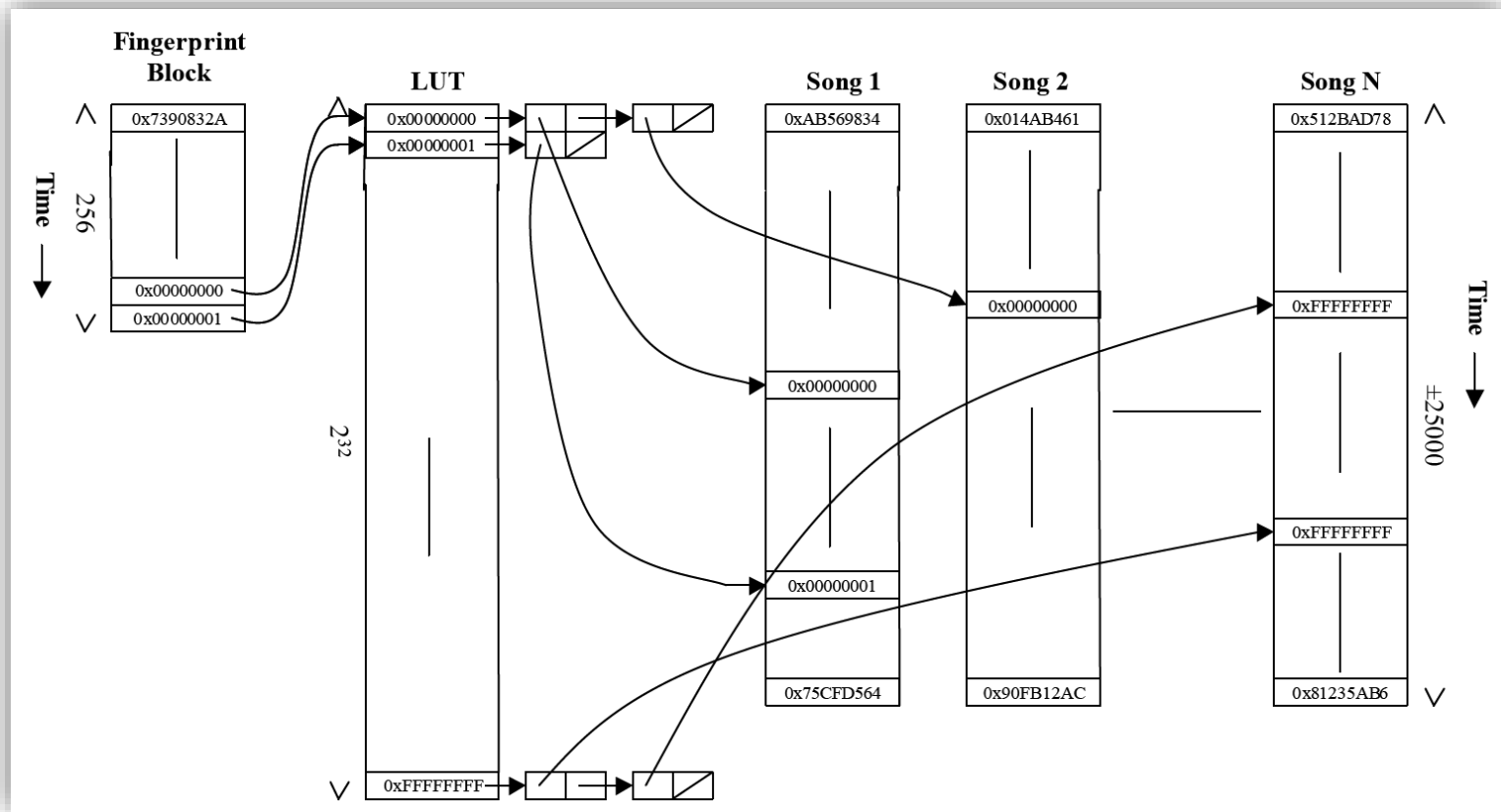


Fig 12. Fingerprint database layout



## 7.1.2.2 Shazam 구현 방법

### ■ Robust constellation 생성

- ◆ FFT을 이용, 시간-주파수 공간으로 변환 (Fig 1.)
- ◆ 매 frame마다 frequency bin으로부터 peak point들을 검출
- ◆ 검출된 Peak point들로부터 sparse set 생성 (Fig 2.)
  - Peak point들은 왜곡이나 잡음에 의한 변형에 강한 장점을 가짐
  - 동일한 두 음악의 경우, 서로 같은 시간 차와 주파수 차를 보이는 peak pair 가 많을 것이라는 가정으로부터 hash를 생성

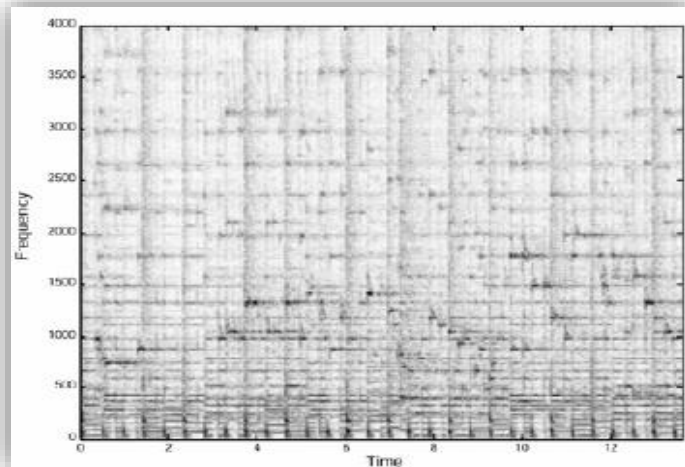


Fig 1. Spectrogram

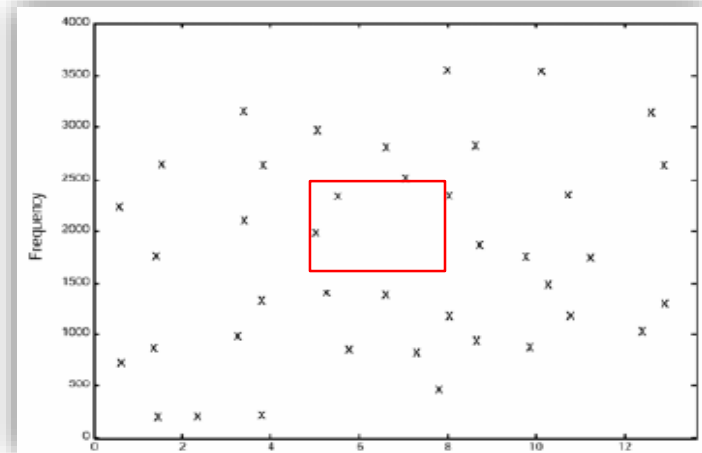


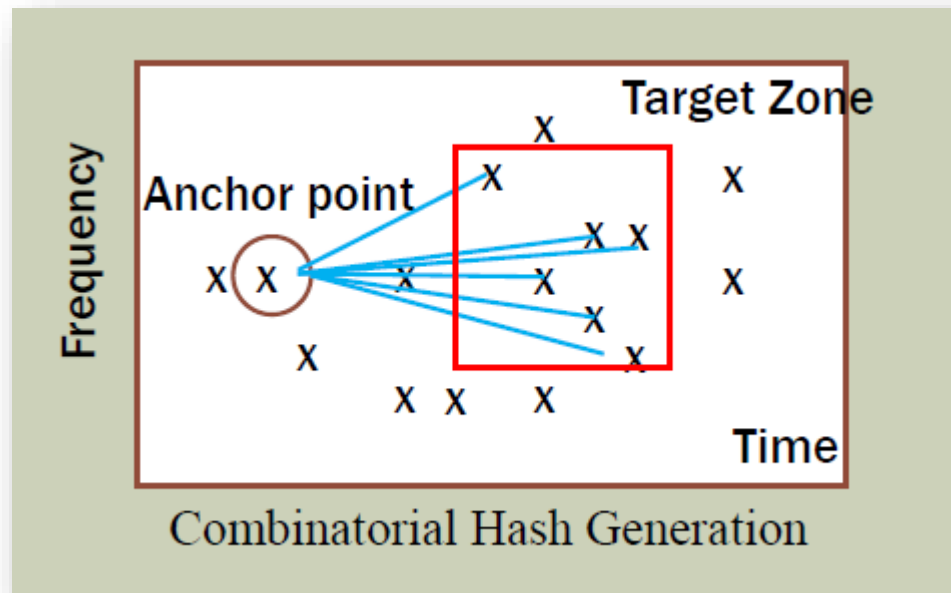
Fig 2. Constellation map



## 7.1.2.2 Shazam 구현 방법

### ■ Fast combinatorial hashing

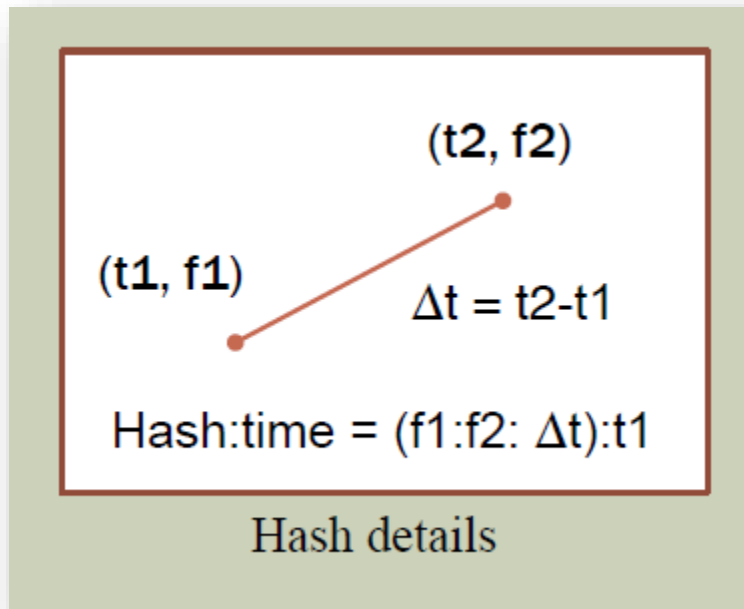
- ◆ 앵커포인트를 설정하고 타겟 지역의 각 포인트에 대해 pair 형성
  - 형성된 pair로부터 두 point의 주파수와 시간차 ( $\Delta t$ )에 해당하는 hash를 생성 (Fig 3.)
- ◆ 하나의 hash( $[f1:f2:\Delta t]$ )는 32bit로 구성( $f1$ : 10bit,  $f2$ : 10bit,  $\Delta t$ : 10bit)
- ◆ Hash 값에 해당 앵커포인트의 time offset ( $t1$ )을 연결 (Fig 4.)



## 7.1.2.2 Shazam 구현 방법

### ■ Fast combinatorial hashing (Cont.)

- ◆ DB 구성을 위해 각 트랙마다 pair set을 만들고 hash에 트랙 ID를 부여
- ◆ one pair = 64bit (32bit for hash, 32bit for time offset and track ID)
- ◆ 앵커포인트의 수와 타겟 지역의 크기에 따라 pair의 수가 결정되는데 이것은 인식율과 DB size의 tradeoff로 결정 됨



## 7.1.2.2 Shazam 구현 방법

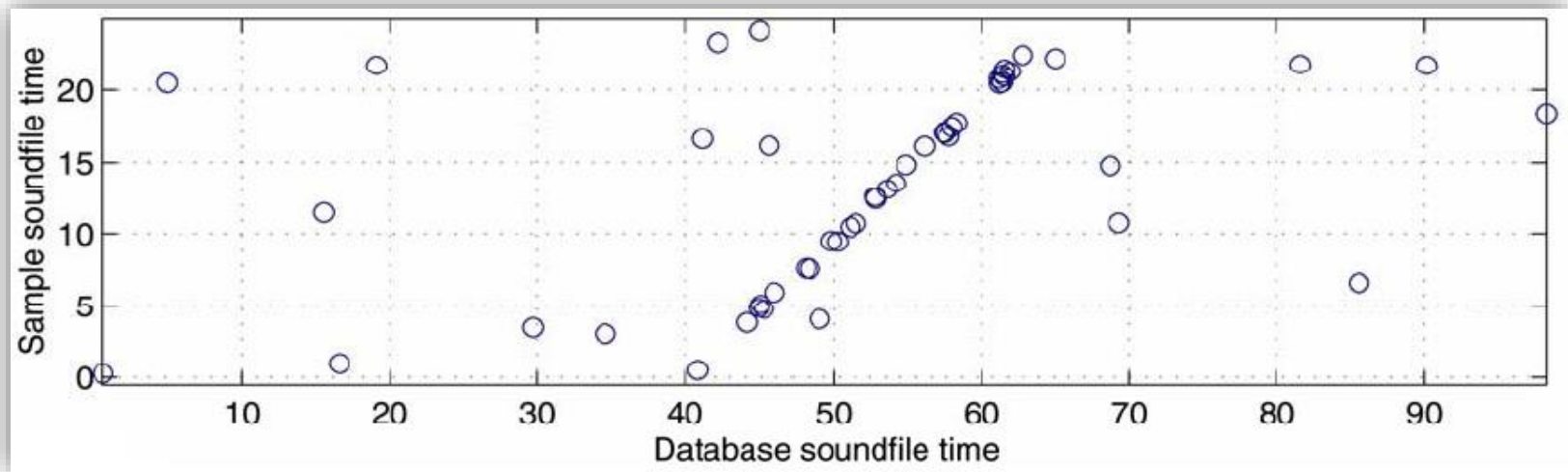
### ■ Searching

- ◆ 녹음된 샘플에서 동일한 방식으로 hash set을 생성
- ◆ 샘플의 hash 값들에 대해서 다음을 수행
  - DB에서 일치하는 hash를 탐색
  - 탐색된 hash의 time offset과 trackID를 검출
  - trackID 별로 bin을 형성 (DB 해시의 time offset과 샘플 해시의 time offset 값의 pair를 저장)

## 7.1.2.2 Shazam 구현 방법

### ■ Searching (Cont.)

- ◆ 위에서 수행된 작업이 완료되면 형성된 trackID bin들을 스캔
  - 각 bin에서 time pair set을 사용하여 scatter plot을 생성
  - 일치하는 파일의 경우, 일치하는 특성은 반드시 시작위치로부터 비슷한 상대 offset위치에 나타나며 Fig 5.와 같이 diagonal하게 일치하는 패턴이 존재



**Fig 5. Scatter plot of matching hash location: Diagonal Present**

## 7.1.2.2 Shazam 구현 방법

### ■ Scoring

- ◆ Scatter plot의 각 point의 x좌표(DB hash의 time offset)과 y좌표(샘플 hash의 time offset)의 차를 히스토그램으로 표현
- ◆ Fig 7.과 같이 동일한 차이를 포인트 수가 score가 됨
- ◆ TrackID bin당 time pair의 수가 적기 때문에 스캐닝 프로세스는 micro second 이하의 시간이 소요

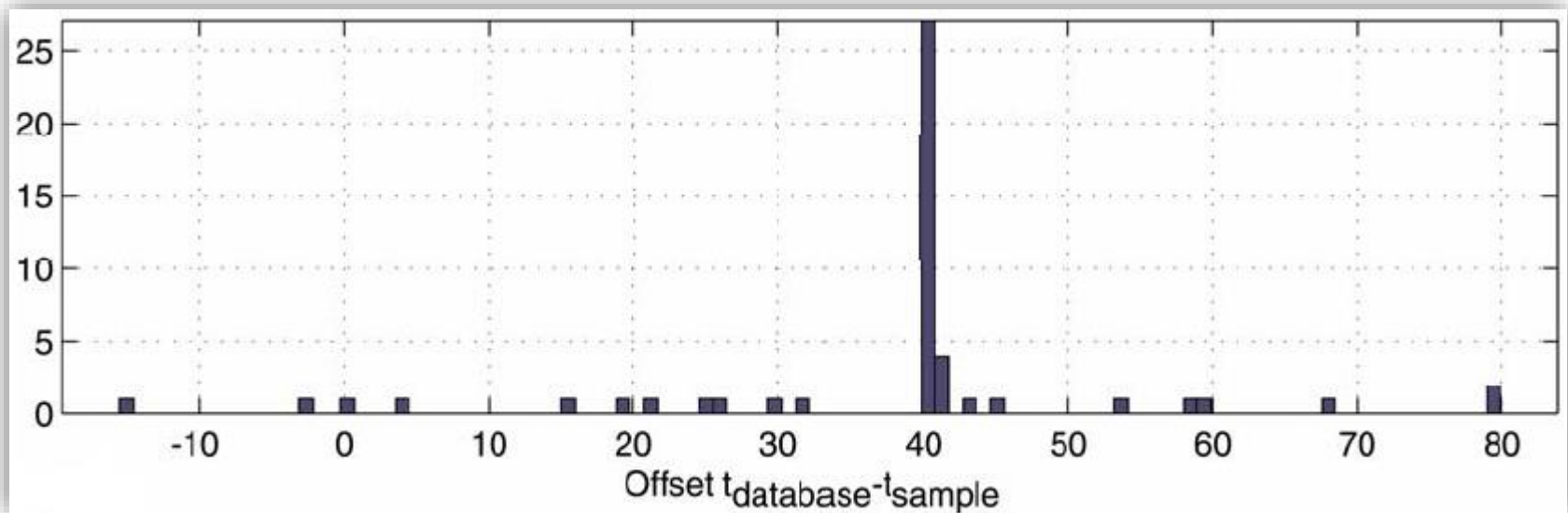


Fig 7. Histogram of difference of time offsets: signals match

## 7.1.2.2 Shazam 구현 방법

### ■ 실험 결과

- ◆ 1만곡 DB에서 5, 10, 15초 샘플 인식 테스트
  - 테스트 곡수: 250곡
- ◆ 실험환경
  - noisy pub (정확한 기술은 없으나 noisy pub에서 마이크를 이용해 녹음한 샘플로 테스트한 것으로 판단)

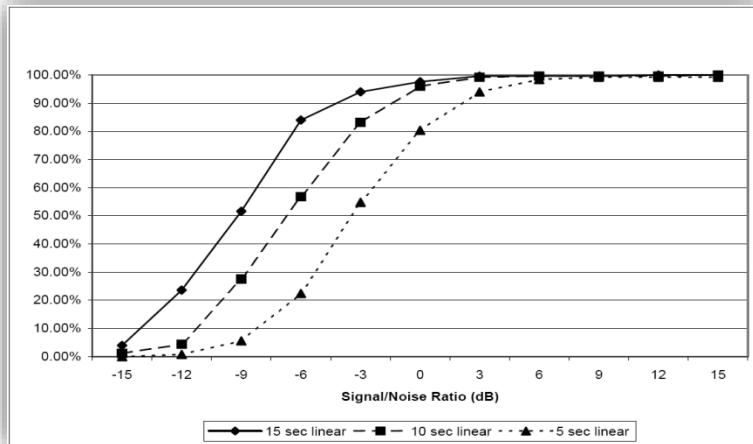


Fig 7. Recognition rate – Additive noise

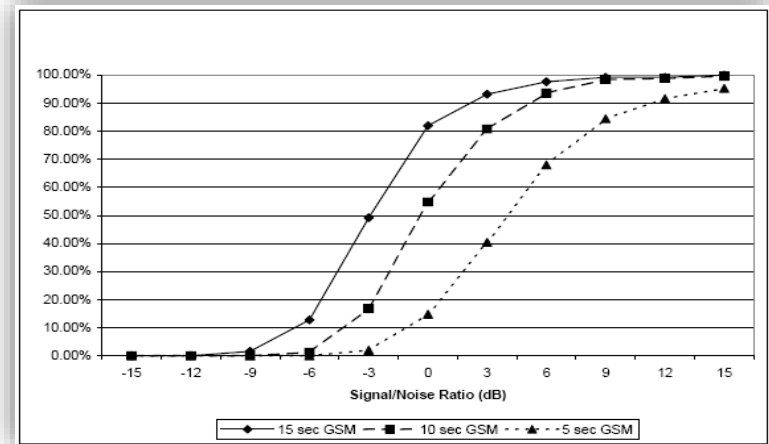


Fig 7. Recognition rate – Additive noise + GSM compression

## 7.1.2.2 Shazam 구현 방법

### ■ 구현시 결정사항

- ◆ Peak point 결정시 범위
- ◆ 앵커포인트 결정 방법
  - Peak point의 일부가 앵커포인트가 됨
- ◆ Target zone 설정 범위
- ◆ 앵커포인트와 target zone 사이의 offset
- ◆ Target zone 내의 peak point중에서 앵커포인트와의 pair에 반영되는 peak point 수 결정



## 7.1.2.3 Audio fingerprint 음악인식 알고리즘 결론

### ■ Grace note와 Shazam구현방식의 차이점

#### ◆ Gracenote

- 연속된 오디오 신호를 sub-fingerprint의 열로 저장
- 인식 단위
  - fingerprint block : 256개의 sub-fingerprint
- Look up table을 해시함수를 이용해 구현
- 일정한 입력길이와 인식 소요시간이 장점

#### ◆ Shazam

- 오디오 신호의 피크 특성을 특징으로 추출
- 피크 정보는 잡음에 강인하지만 fingerprint를 생성하기 위한 입력 길이가 길어져 인식 소요 시간이 오래 걸리는 단점이 있음

## 7.1.3. Query-By-Humming (QBH) 방식

### ■ Query-by-humming (QBH) 시스템

- ◆ P. Bryan, S. Jonah, and B. William, "Name That Tune: A Pilot Study in Finding a Melody From a Sung Query," Journal of the American Society for Information Science and Technology, Feb. 2004, pp.283-300.
- ◆ P. Bryan, "Finding Structure in Audio for Music Information Retrieval," IEEE Signal Processing Magazine, May 2006, pp.126-132.

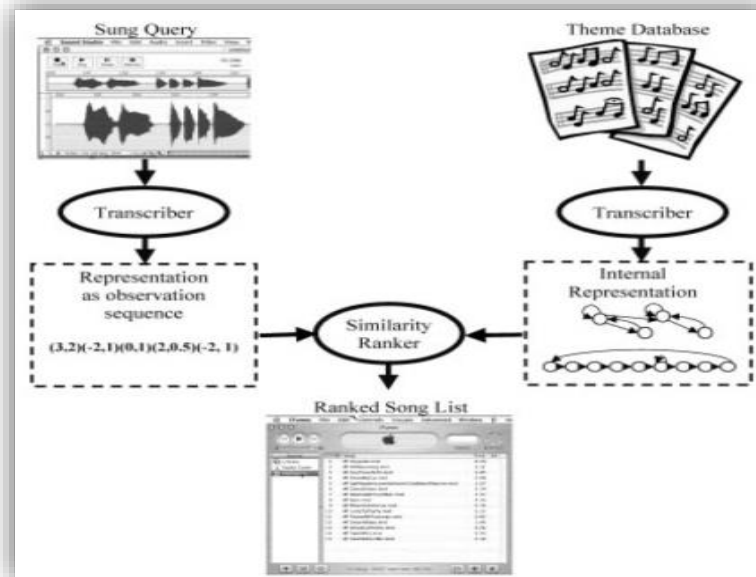


Fig 13. System diagram

## 7.1.3.1 QBH 구현 방법

- 음악을 note열로 변환하여 표현
  - ◆ Pitch-tracking 을 이용하여 프레임마다 pitch 값을 얻음
  - ◆ Pitch 값은 양자화를 거쳐 가장 가까운 note 열로 변환

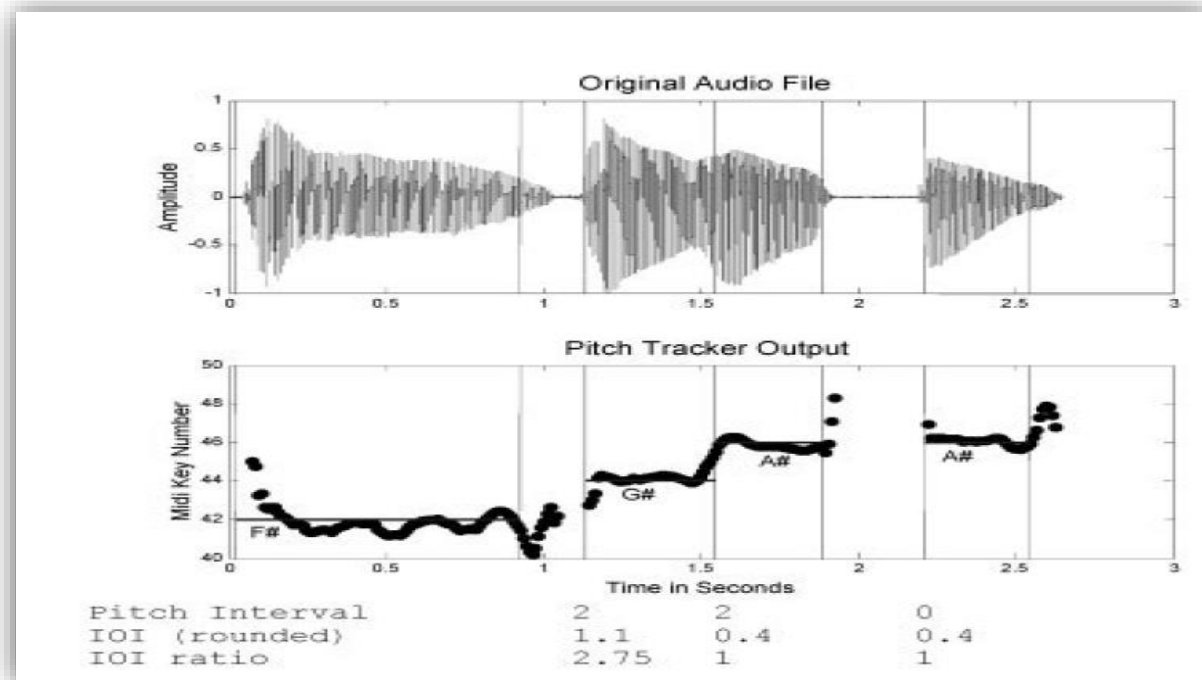


Fig. A transcription of a sung query ([8])

## 7.1.3.1 QBH 구현 방법

### ■ 검색 알고리즘

- ◆ Dynamic programming 방식의 Global Alignment Algorithm을 사용

### ■ Global Alignment Algorithm

- ◆ 질의된 음악내의 note열:  $Q$
- ◆ DB내의 하나의 음악의 note열:  $T$
- ◆  $|Q|+1$  개의 행과  $|T|+1$  개의 열로 이루어진 AlignScore 행렬 생성
- ◆  $AlignScore(i, j)$  :  $Q$ 의 첫번째 note  $q_1$ 부터  $i$ 번째 note  $q_i$ 와  $T$ 의 첫번째 note  $t_1$ 부터  $j$ 번째 note  $t_j$ 까지 가장 잘 정렬된 score값

## 7.1.3.1 QBH 구현 방법

### ■ Global Alignment Algorithm

- ◆  $AlignScore(0, 0)$ 의 값을 0으로 초기화
- ◆  $(0, 0)$ 을 기준으로 방사형으로 Fig 15.의 식들을 이용해  $alignScore$  배열의 모든 셀의 값을 계산
- ◆  $(|Q|, |T|)$ 에서 부터  $(0, 0)$ 의 방향(backward)으로 방사형으로 이동하며 최적의 경로를 구함
- ◆  $(|Q|, |T|)$ 의 score값이 가장 큰 경로 값을 가지는 음악이 입력음악과 매치가 되는 target음악 이라고 봄

## 7.1.3.1 QBH 구현 방법

### ■ Global Alignment Algorithm (Cont.)

$AlignScore(i, j)$

$$= \max \begin{cases} AlignScore(i-1, j-1) + matchScore(q_i, t_j) \\ AlignScore(i-1, j) - skipPenaltyTarget(t_j) \\ AlignScore(i, j-1) - skipPenaltyQuery(q_i) \end{cases}$$

$$matchScore(q_i, t_j) = \begin{cases} 2, & \text{if } q_i = t_j \\ -2, & \text{otherwise} \end{cases}$$

$$\begin{aligned} skipPenaltyQuery(q_i) &= 1 \\ skipPenaltyTarget(t_j) &= 1 \end{aligned}$$

Target \ Query		G	A	B	B
	0	-1	-2	-3	-4
G	-1	2	1	0	-1
D	-2	1	0	-1	-2
A	-3	0	3	2	-1
C	-4	-1	2	1	0
B	-5	-2	1	4	3

Fig 17. Alignment matrix

## 7.1.3.1 QBH 구현 방법

### ■ 구현시 결정사항

- ◆ Note의 개수 (128개로 충분한가?)
- ◆ 사람이 노래를 불렀을 때, 완벽히 일치하는 note열이 생성되기 어려움
  - 여성가수가 부른 노래를 남자가 부른다면?
  - Note열 대신 pitch 변화를 모델링 해야 함
- ◆ Pitch가 여러 개인 경우 안정적으로 pitch 가 추출되는가?
  - 예: 여러 악기가 동시에 연주될 때
- ◆ Note가 다른 경우, 부분 score 배정 문제
  - 인용한 논문에 의하면 서로 다른 note에 대한 score는 동일하게 -2  
이므로 부분 score의 책정이 필요함



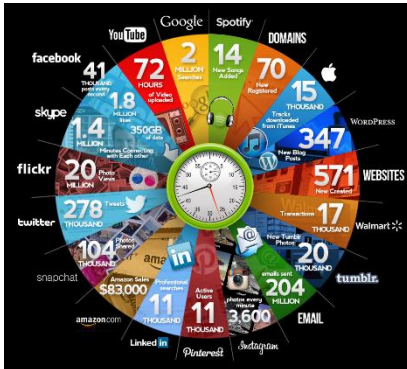
## 7.2.1 오디오 이벤트 분류 - 필요성

총격 scene?



총 이미지 분류

총 소리 분석



수 많은 멀티미디어  
콘텐츠 분석에

이미지 분류와 함께  
필수적인 요소

자동차 추격  
scene?



자동차 이미지 분류

자동차 소리 분석

## 7.2.1 오디오 이벤트 분류 - challenge

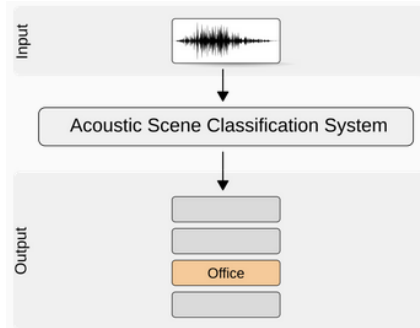
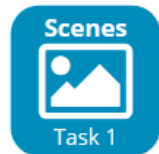
### ■ DCASE Challenge

- ◆ Detection and Classification of Acoustic Scenes and Events
- ◆ 일상적인 환경에서 수집한 오디오 신호를 대상으로 다양한 오디오 이벤트 분류 방법론을 검증
- ◆ 오디오 신호의 종류 및 구분 방법 등 evaluation 조건에 따라 4개의 task로 구분

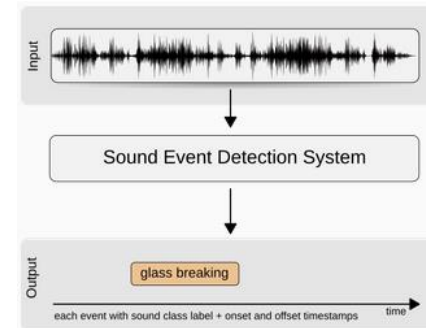
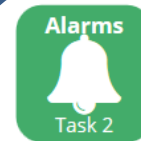
# 7.2.1 오디오 이벤트 분류 - challenge

## ■ DCASE Challenge 2016

### ◆ 2016 evaluation의 task별 내용 및 결과



- Acoustic scene classification
- 소리를 듣고 현재 어디에 있는지 분류 하는 문제
- 총 15개 클래스 (공원, 도로, 바닷가 등)
- 30초 길이의 segment 분류
- 약 25기관 참여
- 상위 5개 기관의 분류성공률  
Accuracy: 85.4% ~ 89.7%

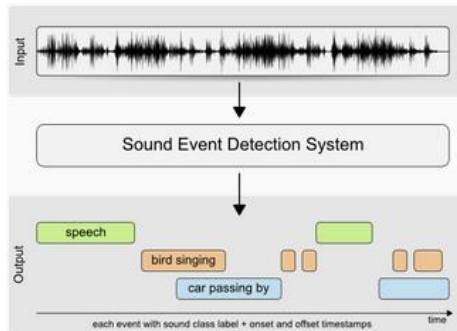


- Sound event detection in synthetic audio
- 사무실 조용한 환경에서 발생하는 소리의 구간을 검출하는 문제
- 총 11개 클래스 (기침, 문소리, 휴대폰 등)
- 1초 단위로 클래스별 on/off 정확도
- 약 10개 기관 참여
- 상위 5개 기관의 검출 성공률  
Accuracy: 41% ~ 67%

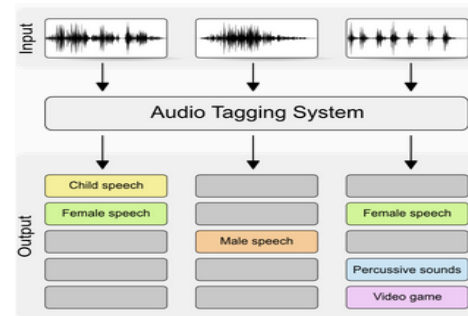
# 7.2.1 오디오 이벤트 분류 - challenge

## ■ DCASE Challenge 2016

### ◆ 2016 evaluation의 task별 내용 및 결과



- Sound event detection in real life audio
- 실 생활 환경에서 발생하는 소리들의 구간을 검출하는 문제 (중복 클래스 가능)
- 총 18개 클래스 (새소리, 자동차 지나가는 소리, 사람말소리, 바람소리 등)
- 1초 단위로 클래스별 on/off 이 모두 맞아야 correct 처리
- 약 10개 기관 참여
- 상위 5개 기관의 검출 성공율  
Accuracy: 9% ~ 20%



- Domestic audio tagging
- 손쉽게 오디오 태깅 할수 있는 방법론을 제시

## 7.2.1 오디오 이벤트 분류 - challenge

### ■ DCASE Challenge 2017

- ◆ 2017 evaluation에서 Google의 AudioSet data를 이용한 large-scale audio event classification task가 추가되었음
- ◆ 대규모 dataset인 AudioSet에서 Auto driving에 관련한 일부 클래스 17개를 선정하여 evaluation 진행

#### Warning sounds:

- Train horn (441)
- Truck horn (407)
- Car alarm (273)
- Reversing beeps (337)
- Ambulance siren (624)
- Police siren (2,399)
- Fire truck siren (2,399)
- Civil defense siren (1,506)
- Screaming (744)

#### Vehicle sounds:

- Bicycle (2,020)
- Skateboard (1,617)
- Car (25,744)
- Car passing (3,724)
- Bus (3,745)
- Truck (7,090)
- Motorcycle (3,291)
- Train (2,301)



## 7.2.1 오디오 이벤트 분류 - dataset

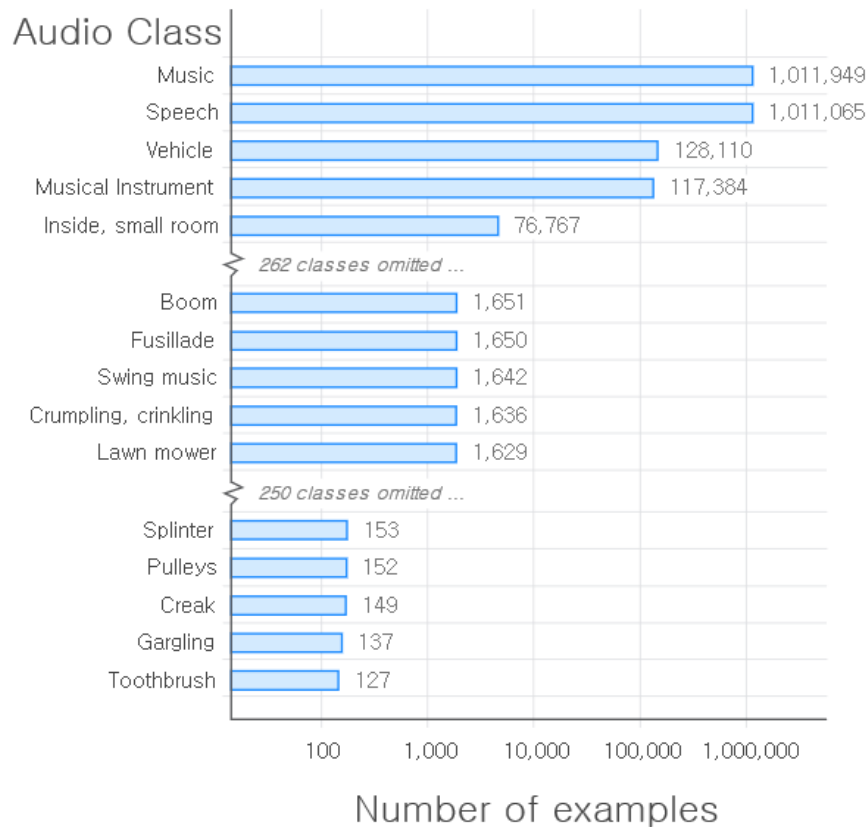
### ■ AudioSet by Google

- ◆ A large-scale dataset of manually annotated audio events
- ◆ Wikipedia, Wordnet 등으로부터 sound event를 새로 추가하거나 관계를 수정하여 오디오 온톨로지를 구축
- ◆ 구축된 온톨로지의 sound event 들을 youtube 영상으로부터 사람이 수작업으로 태깅
- ◆ 총 527 소리 label을 2백만개의 동영상으로부터 태깅

# 7.2.1 오디오 이벤트 분류 - dataset

## ■ AudioSet by Google

### ◆ AudioSet의 온톨로지 및 audio class 분포



#### Human sounds

- Human voice
- Whistling
- Respiratory sounds
- Human locomotion
- Digestive
- Hands
- Heart sounds, heartbeat
- Otoacoustic emission
- Human group actions

#### Source-ambiguous sounds

- Generic impact sounds
- Surface contact
- Deformable shell
- Onomatopoeia
- Silence
- Other sourceless

#### Animal

- Domestic animals, pets
- Livestock, farm animals, working animals
- Wild animals

#### Sounds of things

- Vehicle
- Engine
- Domestic sounds, home sounds
- Bell
- Alarm
- Mechanisms
- Tools
- Explosion
- Wood
- Glass
- Liquid
- Miscellaneous sources
- Specific impact sounds

#### Music

- Musical instrument
- Music genre
- Musical concepts
- Music role
- Music mood

#### Natural sounds

- Wind
- Thunderstorm
- Water
- Fire

#### Channel, environment and background

- Acoustic environment
- Noise
- Sound reproduction



## 7.2.2.1 기존 오디오 이벤트 분류

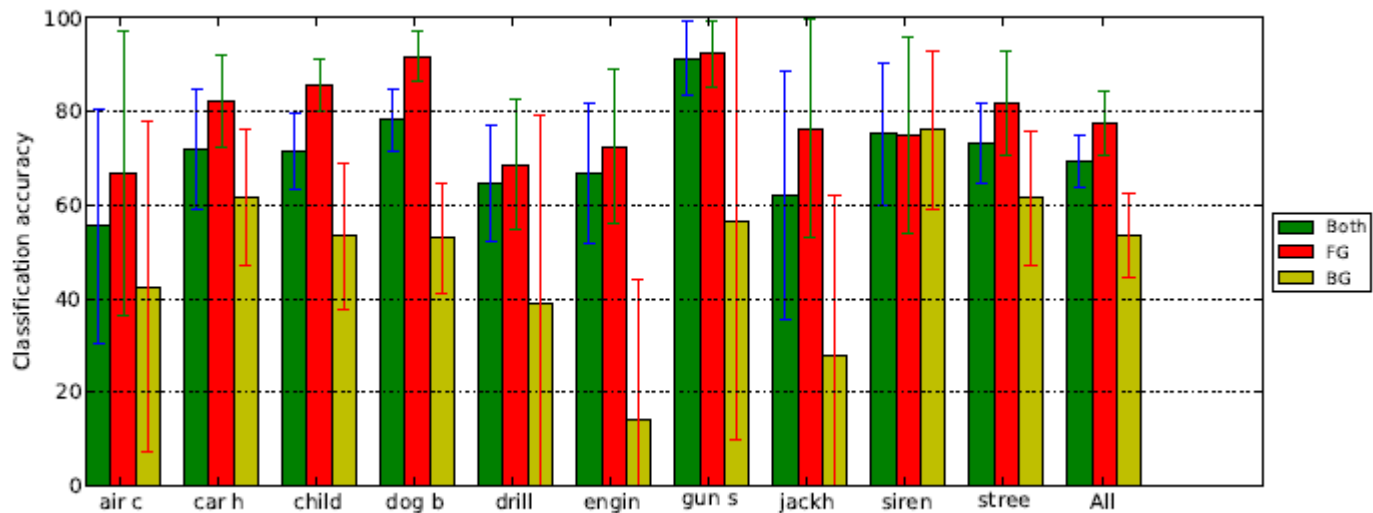
### ■ 기존 연구

- ◆ MFCC / ZCR / Spectral Flux 등 어떤 feature를 사용할 것인지에 대한 연구가 주를 이룸
- ◆ Classifier로는 주로 GMM / SVM 등이 사용 됨
- ◆ Category 수가 제한적임 (Speech / Music / 기타 등)

## 7.2.2.1 기존 오디오 이벤트 분류

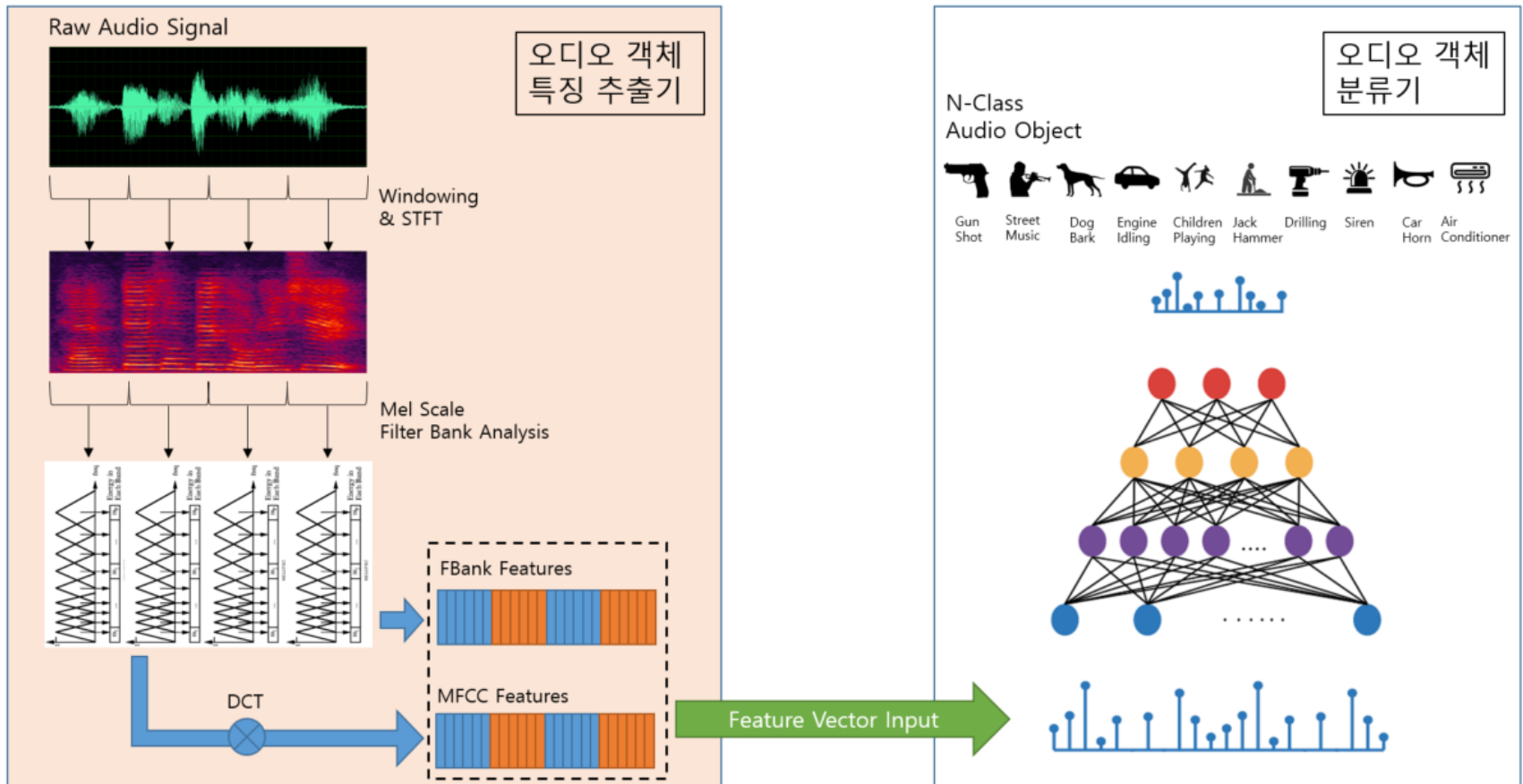
### ■ [J. Salamon, 2015]

- ◆ Classifier: SVM
- ◆ Feature: MFCC (Total 225-dimension)
- ◆ Dataset: UrbanSound8K dataset (약 8-9 시간)
- ◆ 최대 성능: 10-Class, Sample Accuracy – 약 70%



## 7.2.2.2 DNN 기반 오디오 이벤트 분류

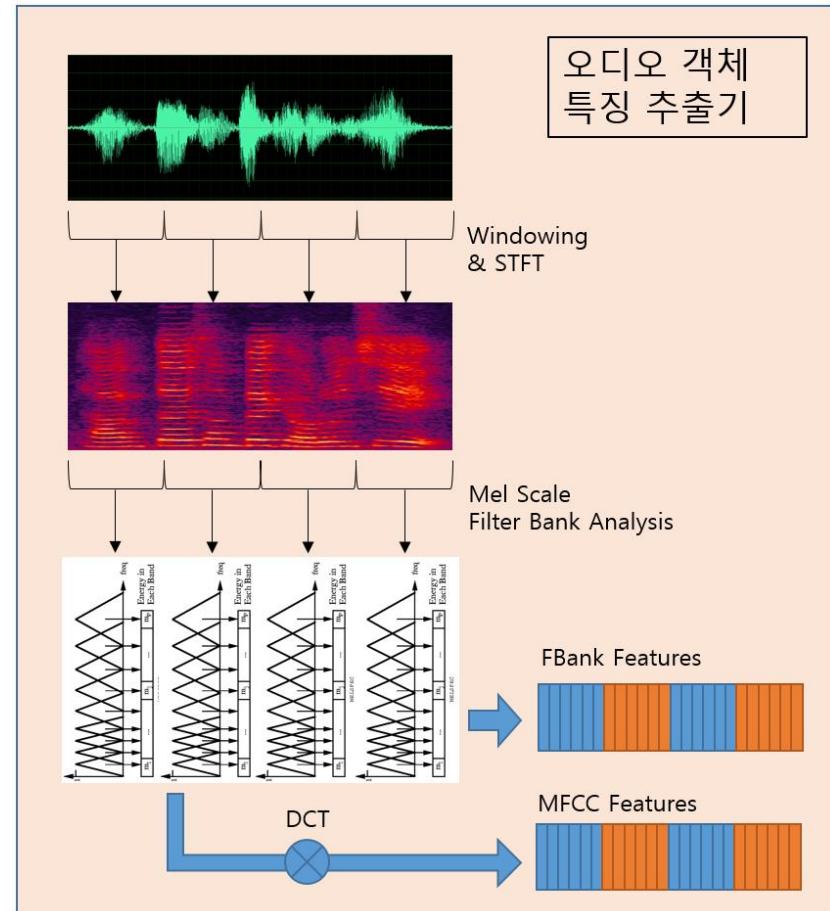
### ■ 오디오 객체 특징 추출기 및 분류기 구조



## 7.2.2.2 DNN 기반 오디오 이벤트 분류

### ■ 오디오 객체 특징 추출

- ◆ 입력된 오디오 신호에 대하여 10ms 단위로 이동하는 20ms 길이의 Hamming window를 씌움
- ◆ Window 별로 STFT 및 Filter Bank 분석을 통한 특징 값을 생성
- ◆ Context 정보를 고려하여 좌/우 N개의 윈도우로부터의 특징값을 합하여 입력벡터로 구성



## 7.2.2.3 CNN 기반 오디오 이벤트 분류

### ■ CNN 기반 오디오 이벤트 분류 구조

- ◆ 매 프레임 별로 Fbank 특징 vector를 생성 후
- ◆ 연속된 프레임들로 추출된 Fbank 특징벡터를 하나의 이미지 data로 간주하여 CNN의 입력 벡터로 사용함
- ◆ 최근 오디오 분류 challenge에서 상위권을 차지하고 있음

