

음성, 음악, 소음구별 및 NAVER음성인식 연결 모델

1. 프로젝트 개요

Siri야~ NUGU야~ Gini야~ 하고 부르는 음성인식 엔진들을 보면 신기하기도 하고, 이를 활용하면 훨씬 편리한 서비스를 만들 수 있을 것 같은데 어떻게 하면 이런 기능을 이용해 볼 수 있을까? 단순히 음성 인식 기능만 있으면 되는 것이 아니라, 일상에서는 음악, 소음도 있을 텐데 어떻게 하면 음성을 구분해서 인식할 수 있을까? 이런 궁금증을 해소해 볼 수 있는 간단한 프로젝트를 진행해 보고자 한다.

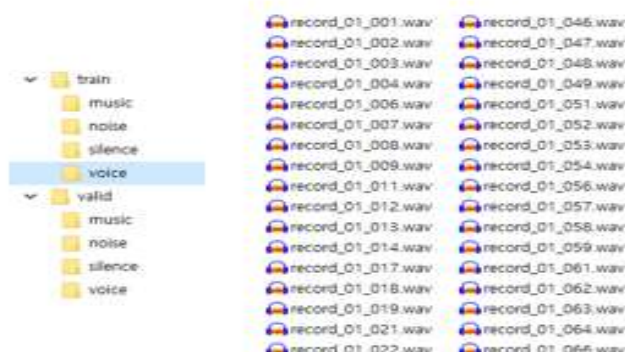
2. 프로젝트 목적

본 프로젝트에서는 음성(Voice)을 음악(Music)과 소음(Noise)으로부터 구별하는 모델을 만들어보고, 구별된 음성에 대해 NAVER음성인식 엔진에 연결하여 어떤 말인지 인식시켜 보는 것을 목표로 한다.

3. 프로젝트 내용

1) 데이터 수집

- 2초 단위 음성, 음악, 소음 데이터를 준비한다.
- 음성은 직접 녹음한 파일을 이용할 수도 있고, 다양한 음성과 사운드를 수집하기 위해 Youtube에서 적합한 파일을 찾아 mp3형태로 다운로드 받은 후, 이를 wav형태로 변환하여 이용할 수도 있다.
- 소음 데이터로는 생활소음을 녹음해서 이용할 수도 있고, Youtube에서 다양한 자연의 소리를 다운받아서 데이터로 만들어볼 수도 있다.

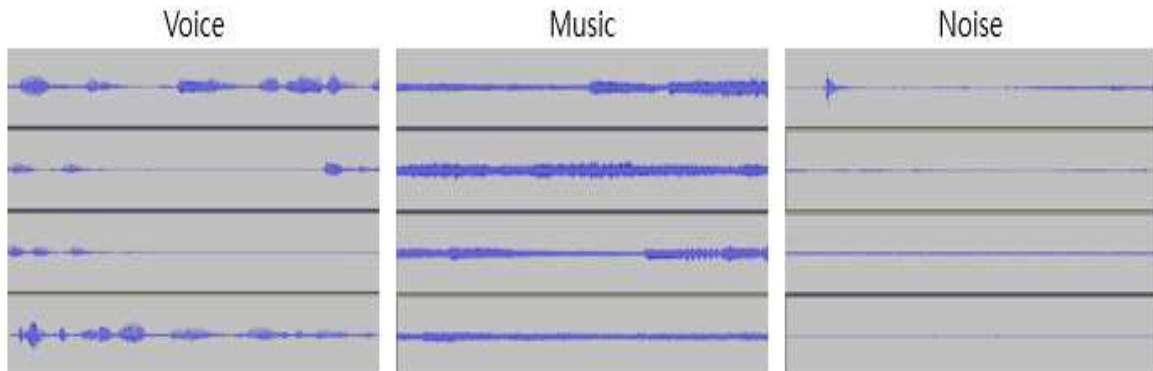


[그림 1] 다양한 사운드 데이터 구성 (voice, music, noise)

- 수집한 데이터는 Train : Valid = 8 : 2로 나누어 학습 데이터셋을 구성한다.

2) Sound 파형

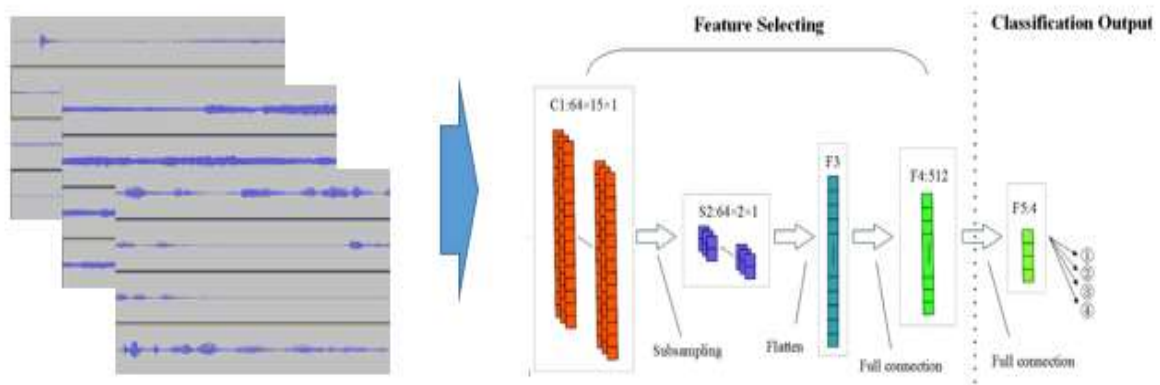
- 음성(Voice)과 음악(Music), 소음(Noise) 파형을 살펴보면 그 형태에 차이가 있음을 알 수 있다.



[그림 2] 음성, 음악, 소음의 파형 모습. 음성은 상대적으로 누운 항아리 모양(중간이 불룩한 형태)을 하고 있고, 음악(연주곡 중심)은中间的 두꺼운 부분이 좀 더 길게 지속되는 벽돌모양 파형을 하고 있다. 소음은 순간적으로 강한 소리가 튀듯이 발생하는 특징이 있어 구별된다.

3) 파형 학습하기 (Tensorflow 1D CNN)

- Tensorflow 1D-CNN 모델을 구성하여 다양한 사운드 파형을 학습한 후 카테고리별로 분류하는 classification model을 구성해 보았다.

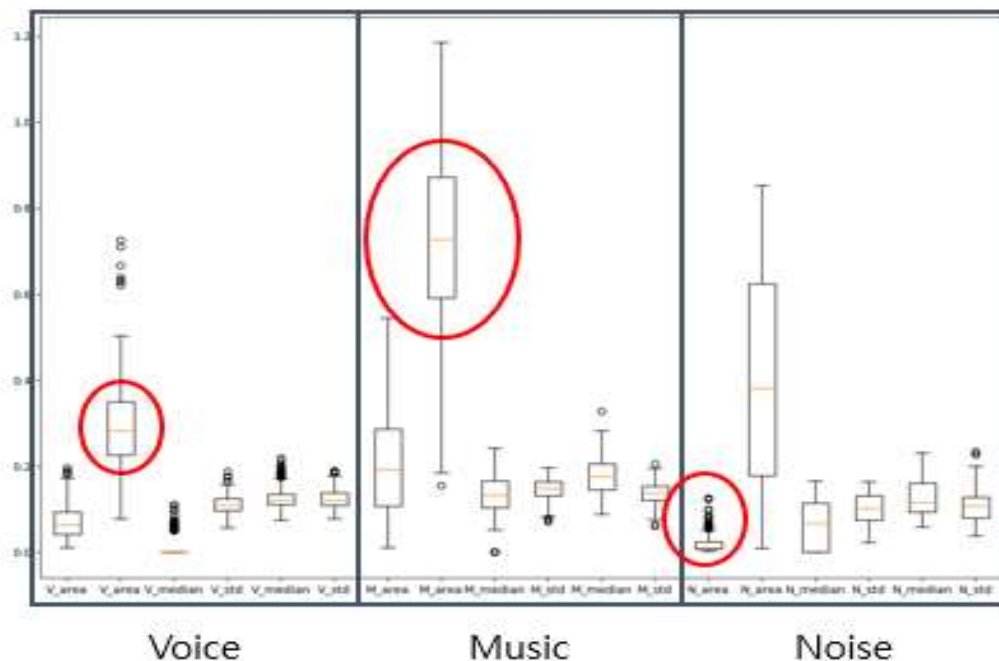


[그림 3] 사운드 파형을 1D-CNN으로 분석하여 특징을 추출하고, 이 특징들을 학습하여 사운드를 분류하는 classification모델 구성도

- 1D-CNN모델은 1차원 signal data에 적용할 수 있는 모델로, 해당 사운드만의 독자적인 특성을 찾아낼 수 있다.

4) Rule-based model

- 음성, 음악, 소음 파형이 많이 다르기 때문에 이론적으로는 1D-CNN모델로 충분히 구분이 가능할 것으로 보이나, 데이터가 충분하지 않아서인지 성능이 아주 높게 나오지는 않았다.
- 데이터를 더 수집할 수도 있지만, 여기서는 Rule-based method도 살펴보았다.
- 파형 크기를 Normalize시키면, 음성보다 음악소리가 평균값이 높게 나타나는데 이는 음악이 음성보다 짝 찬(평균음량이 큰) 소리로 구성되어 있기 때문이다.
- 소음은 순간적으로 나는 소리인 경우가 많아서 파형의 면적값이 매우 작게 나타난다.
- 이러한 특징들을 구별할 수 있는 함수를 만들어 rule-based 방식으로 구별할 수도 있다.



[그림 4] 음성, 음악, 소음 사운드 데이터의 다양한 특성.
(파형면적_Origin, 파형면적_Normalized, 중앙값, 표준편차)

5) 구분결과

- Rule-based 방식을 적용할 경우, 다른 소리로부터 음성을 구별하는 정확도가 약 91.7% (88/96, validation set 기준)로 나오며, 음성을 잘 구별할 수 있음을 알 수 있다.

6) 음성인식 엔진 연결

- 개인 음성인식 엔진을 새로 만들기는 용이하지 않을 것이며, NAVER CLOVA의 STT(SpeechToText) Open API를 이용해 보았다.



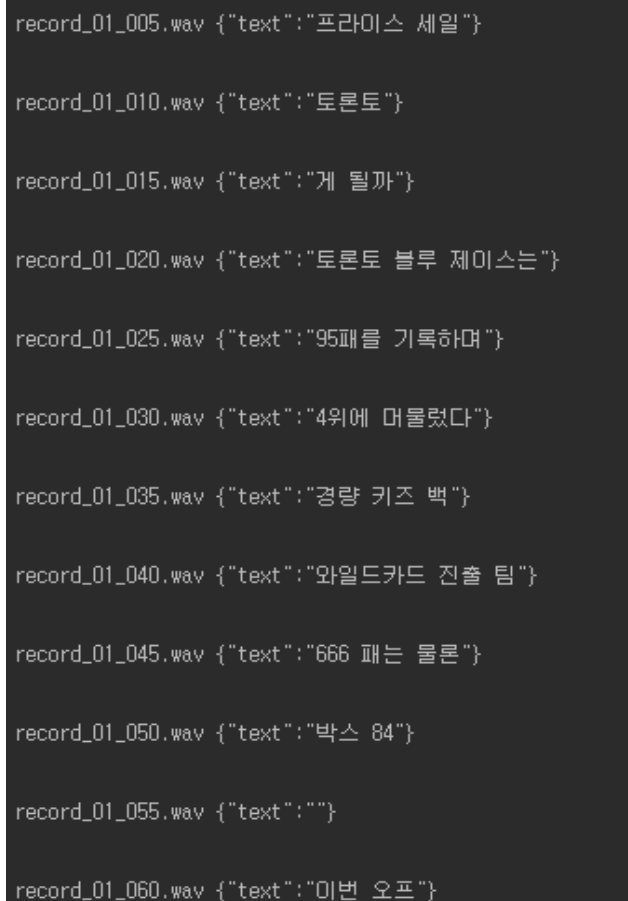
```

JAVA  PHP  JS  PYTHON  C#

import sys
import requests
client_id = "YOUR_CLIENT_ID"
client_secret = "YOUR_CLIENT_SECRET"
lang = "Kor" # 언어 코드 (Kor, Jpn, Eng, Chn)
url = "https://naveropenapi.apigw.ntruss.com/recog/v1/stt?lang=" + lang
data = open('음성 파일 경로', 'rb')
headers = {
    "X-NCP-APIGW-API-KEY-ID": client_id,
    "X-NCP-APIGW-API-KEY": client_secret,
    "Content-Type": "application/octet-stream"
}
response = requests.post(url, data=data, headers=headers)
rescode = response.status_code
if(rescode == 200):
    print(response.text)
else:
    print("Error : " + response.text)
```

[그림 5] NAVER CLOVA STT 모듈의 Python API

- 5번에서 음성으로 인식된 wav파일에 한해 STT를 진행하였으며, 다음과 같은 결과를 얻었다.



```

record_01_005.wav {"text": "프라이스 세일"}

record_01_010.wav {"text": "토론토"}

record_01_015.wav {"text": "게 될까"}

record_01_020.wav {"text": "토론토 블루 제이스는"}

record_01_025.wav {"text": "95패를 기록하며"}

record_01_030.wav {"text": "4위에 머물렀다"}

record_01_035.wav {"text": "경량 키즈 백"}

record_01_040.wav {"text": "와일드카드 진출 팀"}

record_01_045.wav {"text": "666 패는 물론"}

record_01_050.wav {"text": "박스 84"}

record_01_055.wav {"text": ""}

record_01_060.wav {"text": "미번 오프"}
```

[그림 6] NAVER CLOVA STT 인식 결과

4. 주요 기술

- 언어 : Python
- 모델/패키지 : 1D-CNN, librosa, requests, NAVER STT api
- 프레임워크 : tensorflow1.15

5. 주요 소스코드

1) 음성 여부 파악 로직 (rule-based 방식)

```
# ----- 연결 앱 -----
# Read wav and plot signal and fft
# ori_area: 0.02 * 10000 이하 제거 -> Noise 제거, 짧은 음성 제거
# norm_area: 0.55 * 10000 이상 제거 -> Music 제거
# norm_median: 0.1 이상 제거
if True:
    AUDIO_PATH = 'dataset/valid/'
    SAMPLE_RATE = 22050
    NFFT = 32768 # 16384
    FREQ_LIMIT = 512
    #
    cnt_correct = 0
    cnt_total = 0
    cnt_correct_voice = 0
    cnt_total_voice = 0
    for dirname, subdirs, files in os.walk(AUDIO_PATH):
        AUDIO_TYPE = dirname.replace('\\', '/').split('/')[-1]
        if AUDIO_TYPE not in ['voice', 'music', 'noise', 'silence']:
            continue
        #
        for audio_file in files:
            #print(AUDIO_TYPE, audio_file)
            sigs, s_rate = librosa.load(dirname + '/' + audio_file, sr=SAMPLE_RATE)
            sigs = np.abs(sigs)
            sigs_norm = sigs / np.max(sigs)
            #
            if True:
                # 0 제거 후 Re-structuring
                sigs_norm[sigs_norm < 0.05] = 0
                sigs_norm = (sigs_norm - np.min(sigs_norm)) / (np.max(sigs_norm) - np.min(sigs_norm))
                sigs_no_zero = []
                for s in range(len(sigs_norm)):
                    sig = sigs_norm[s]
                    if sig != 0:
                        sigs_no_zero.append(sig)
                #
                sigs_restruct = []
                for i in range(44100):
                    sigs_restruct.append(sigs_no_zero[i % len(sigs_no_zero)])
                sigs_restruct = np.array(sigs_restruct)
            #
            pred = 'others'
            # np.sum(sigs_norm) >= 5000 and np.sum(sigs_norm) <= 10000:
            if np.median(sigs_norm) >= 0.00 and np.median(sigs_norm) <= 0.12 \
                and np.std(sigs_norm) >= 0.05 and np.std(sigs_norm) <= 0.15 \
                and np.sum(sigs) >= 200 and np.sum(sigs_norm) <= 5500:
                pred = 'voice'
            #
            #print(audio_file, 'type:', AUDIO_TYPE, ', pred:', pred)
            cnt_total += 1
            if AUDIO_TYPE == 'voice':
                cnt_total_voice += 1
                if AUDIO_TYPE == pred:
                    cnt_correct += 1
                    cnt_correct_voice += 1
            else:
                if pred == 'others':
                    cnt_correct += 1
```

2) NAVER STT api 연결코드

```
client_id = "krc0aufyjdj"
client_secret = "3JdVK6FeBz3I9hRmoiBSBmfjUW3P83jboDY6UxJt"
lang = "Kor" # 언어 코드 (Kor, Jpn, Eng, Chn)
url = "https://naveropenapi.apigw.ntruss.com/recog/v1/stt?lang=" + lang
```

3) 종합테스트_1: 음성여부 파악(rule-based method)

```

# -----
# Read wav and plot signal and fft
# ori_area: 0.02 * 10000 이하 제거 -> Noise 제거, 짧은 음성 제거
# norm_area: 0.55 * 10000 이상 제거 -> Music 제거
# norm_median: 0.1 이상 제거
AUDIO_PATH = 'dataset/valid/'
SAMPLE_RATE = 22050
NFFT = 32768 # 16384
FREQ_LIMIT = 512
#
cnt_correct = 0
cnt_total = 0
cnt_correct_voice = 0
cnt_total_voice = 0
for dirname, subdirs, files in os.walk(AUDIO_PATH):
    AUDIO_TYPE = dirname.replace('\\', '/').split('/')[-1]
    if AUDIO_TYPE not in ['voice', 'music', 'noise', 'silence']:
        continue
    #
    for audio_file in files:
        #print(AUDIO_TYPE, audio_file)
        sigs, s_rate = librosa.load(dirname + '/' + audio_file, sr=SAMPLE_RATE)
        sigs = np.abs(sigs)
        sigs_norm = sigs / np.max(sigs)
        #
        if True:
            # 0 제거 후 Re-structuring
            sigs_norm[sigs_norm < 0.05] = 0
            sigs_norm = (sigs_norm - np.min(sigs_norm)) / (np.max(sigs_norm) - np.min(sigs_norm))
            sigs_no_zero = []
            for s in range(len(sigs_norm)):
                sig = sigs_norm[s]
                if sig != 0:
                    sigs_no_zero.append(sig)
            #
            sigs_restruct = []
            for i in range(44100):
                sigs_restruct.append(sigs_no_zero[i % len(sigs_no_zero)])
            sigs_restruct = np.array(sigs_restruct)
        #
        pred = 'others'
        # np.sum(sigs_norm) >= 5000 and np.sum(sigs_norm) <= 10000:
        if np.median(sigs_norm) >= 0.00 and np.median(sigs_norm) <= 0.12 \
            and np.std(sigs_norm) >= 0.05 and np.std(sigs_norm) <= 0.15 \
            and np.sum(sigs) >= 200 and np.sum(sigs_norm) <= 5500:
            pred = 'voice'
        #

```

4) 종합테스트_2: 음성인식

```

cnt_total_voice = 0
cnt_total = 0
if AUDIO_TYPE == 'voice':
    cnt_total_voice += 1
    if AUDIO_TYPE == pred:
        cnt_correct += 1
        cnt_correct_voice += 1
# -----
# 음성인식
try:
    data = open(dirname + '/' + audio_file, 'rb')
    headers = {
        "X-NCP-APIGW-API-KEY-ID": client_id,
        "X-NCP-APIGW-API-KEY": client_secret,
        "Content-Type": "application/octet-stream"
    }
    response = requests.post(url, data=data, headers=headers)
    rescode = response.status_code
    if (rescode == 200):
        print()
        print(audio_file, response.text)
    else:
        print("Error : " + response.text)
except Exception as ex:
    print(str(ex))
# -----

```