



[DFC-2023-104] Write-Up

작성자	허은정
분석 일자	2024.05.24
작성 일자	2024.05.24
분석 대상	modified.wav original.wav
문서 버전	1.0
작성자 E-mail	dmswjd4315@yonsei.ac.kr

0. 목차

1. 문제3

2. 분석 도구3

3. 환경3

4. Write-Up.....4



5. Flag.....8

6. 별도 첨부9

7. Reference 10



1. 문제

URL	-
문제 내용	<p>Here is an audio file that supposedly added a fake voice intentionally. As a digital forensic investigator, solve the following questions.</p> <p>1) At what time does the fake voice play? (10 points)</p> <p>2) Provide evidence to support the answer. (90 points)</p>
문제 파일	<div>  modified.wav  original.wav </div>
문제 유형	multimedia forensics
난이도	2 / 3

2. 분석 도구

도구명	다운로드 링크	Version
Audacity	https://www.audacityteam.org/	3.5.1

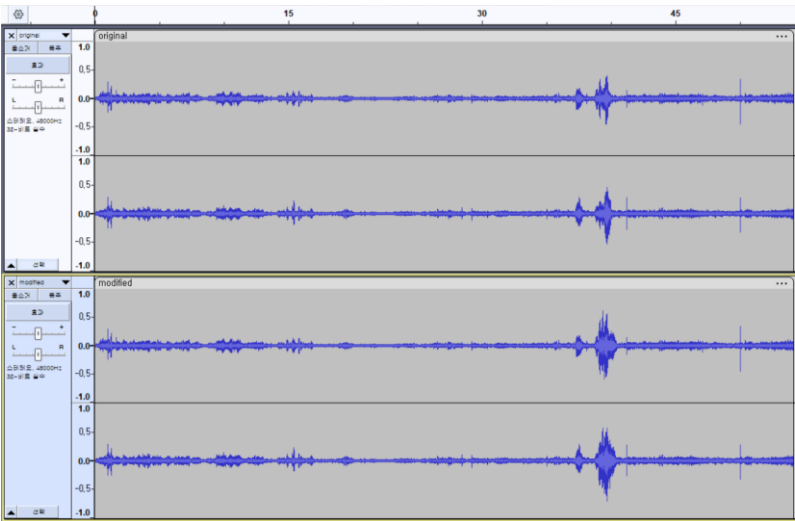
3. 환경

OS
Window 11 64-bit

4. Write-Up

파일명	modified.wav / original.wav
용량	19.8MB / 19.8MB
SHA256	380e91ed5ebe831100a6612d9b108090ece59a6de3857db306300e563a2ab302/ 25bfb194892857031a2b7662bbc9616d2b5bc54ec09311badcd8c4f564ec79a2
Timestamp	2023-06-29 15:12:32 / 2023-06-29 15:23:56

1) At what time does the fake voice play? (10 points)



[사진 1] Audacity로 열어본 파일

Audacity 를 사용하여 'original.wav'와 'modified.wav' 파일을 불러왔다.

두 파일의 관계를 독립적으로 비교한 결과, 'modified.wav' 파일의 약 38 초부터 41 초 사이의 차이점을 발견했다.

파형 비교 및 변조 음성을 검출 하기 위해 STFT(Short-time Fourier Transform)를 사용하여 스트리밍과 시간 정보를 동시에 분석했다.



```
fake audio start at 38.97747401799117 seconds
```



```
fake audio finish at 40.78822027969653 seconds
```

[사진 2] 변조된 음성이 존재하는 부분

STFT 기법을 사용하여 분석 결과를 표현하고 참여하는 방식을 다음과 같이 확인했다: 자세한 풀이 과정은 2 번 문제에 서술하였다.

Fake audio start: 38.97747401799117 초

Fake audio finish: 40.78822027969653 초

2) Provide evidence to support the answer. (90 points).

dfc2023-104_1.py 코드는 STFT 기법을 활용하여 변조된 음성을 검출하기 위해 임계값을 찾아내는 코드이다.

```
# -*- coding: utf-8 -*-
"""DFC2023-104-1.ipynb

Automatically generated by Colab.

Original file is located at
https://colab.research.google.com/drive/1QKYjDfNPC10cQp-8XgAow1AuAPbcU2Rkx

from google.colab import drive
import matplotlib.pyplot as plt
import numpy as np
import librosa

drive.mount('/content/gdrive')

original_audio, sr_original_audio = librosa.load('/content/gdrive/MyDrive/original.wav')
modified_mix_audio, sr_modified_mix_audio = librosa.load('/content/gdrive/MyDrive/modified.wav')

A_original_audio = np.abs(librosa.stft(original_audio))
A_modified_mix_audio = np.abs(librosa.stft(modified_mix_audio))

threshold = 1
time_frame = np.sum(np.abs(A_original_audio - A_modified_mix_audio), axis=0)

plt.plot(time_frame)
plt.title('Time Frame Difference')
plt.xlabel('Time Frame')
plt.ylabel('Difference')
plt.show()

# 임계값을 넘는 인덱스 찾기
indices_where_exceeds_threshold = np.where(time_frame > threshold)[0]

# 변경된 음성이 시작되는 인덱스 찾기
modified_voice_start = indices_where_exceeds_threshold[0] if indices_where_exceeds_threshold.size > 0 else None

# 최대값 및 평균값을 분석하여 적절한 임계값을 threshold 변수에 지정
print(f'Difference modified voice start index: {time_frame[modified_voice_start]}')
print(f'Maximum difference until modified voice start index: {np.max(time_frame[:modified_voice_start])}')
print(f'Average difference until modified voice start index: {np.average(time_frame[:modified_voice_start])}')
```

[사진 3] 임계값을 찾아내는 코드

아래는 코드에 대한 설명이다.

1. 해당 코드는 변조된 음성을 정확히 확인하기 위해 파이썬의 librosa 라이브러리를 사용하여 두 오디오 파일인 original.wav 와 modified.wav 의 주파수 차이를 분석한다.
2. librosa.stft 함수를 사용하여 오디오 데이터를 STFT 로 변환한다. STFT 는 시간-주파수 영역에서 신호를 분석하는 방법이다. np.abs 함수를 사용하여 복소수 스펙트로그램의 절대값을 구한다. 이는 주파수 성분의 크기를 나타낸다.

[WHS-2] .iso

3. 두 오디오 파일의 주파수 차이를 시간 프레임별로 계산합니다. threshold 변수에 임계값을 설정하고, `np.sum(np.abs(A_original_audio - A_modified_mix_audio), axis=0)`를 사용하여 주파수 차이의 합을 계산한다. 이를 통해 시간 프레임별 주파수 차이를 시각화한다.
4. `np.where` 함수를 사용하여 주파수 차이가 임계값을 초과하는 시간 프레임의 인덱스를 찾는다. 이는 변조된 음성 구간을 식별하는 데 도움이 된다.

위 코드에서 threshold(임계값)를 1로 설정하여 실행한 결과는 다음과 같다.

- Difference modified voice start index: 2.230407953262329
- Maximum difference until modified voice start index: 0.6315687894821167
- Average difference until modified voice start index: 0.13858658075332642

이를 통해, 임계값을 1로 설정하여 변조된 음성을 찾기 위해 좋은 임계값이라는 것을 알 수 있다.

dfc2023_104_2.py 는 변조된 음성이 존재하는 시간대를 찾는 코드이다.

```
# -*- coding: utf-8 -*-
"""DFC2023-104-2.ipynb
Automatically generated by Colab.

Original file is located at
https://colab.research.google.com/drive/1Em4tBeeYCNulkFqVqov-Lo6jVxoOmM
"""

from google.colab import drive
import matplotlib.pyplot as plt
import numpy as np
import librosa

drive.mount('/content/gdrive')

original_audio, sr_original_audio = librosa.load('/content/gdrive/MyDrive/original.wav')
fake_mix_audio, sr_fake_mix_audio = librosa.load('/content/gdrive/MyDrive/modified.wav')

A_original_audio = np.abs(librosa.stft(original_audio))
A_fake_mix_audio = np.abs(librosa.stft(fake_mix_audio))

threshold = 1
time_frame = np.sum(np.abs(A_original_audio - A_fake_mix_audio), axis=0)
# 임계값을 넘는 인덱스 찾기
indices_where_exceeds_threshold = np.where(time_frame > threshold)[0]

# 변경된 음성이 시작되는 인덱스 찾기
fake_voice_start = indices_where_exceeds_threshold[0] if indices_where_exceeds_threshold.size > 0 else None

# 시작 시각 계산 (fake voice start 가 None 이 아닐 때만 계산)
if fake_voice_start is not None:
    start_time_seconds = fake_voice_start * (len(original_audio) / sr_original_audio) / len(time_frame)
    print(f'fake audio start at {start_time_seconds} seconds')
else:
    print('Threshold not exceeded')

# 주파수 변화 감지 (종료 시각)
if fake_voice_start is not None:
    indices_where_exceeds_threshold_end = np.where(time_frame[fake_voice_start:] <= threshold)[0]
    fake_voice_end = indices_where_exceeds_threshold_end[0] + fake_voice_start if indices_where_exceeds_threshold_end.size > 0 else None

    # 종료 시각 계산 (fake_voice_end가 None이 아닐 때만 계산)
    if fake_voice_end is not None:
        end_time_seconds = fake_voice_end * (len(original_audio) / sr_original_audio) / len(time_frame)
        print(f'fake audio finish at {end_time_seconds} seconds')
    else:
        print('No end point found within threshold')
else:
    fake_voice_end = None
    print('Threshold not exceeded at start, so end point not calculated')

# 시각화
plt.figure(figsize=(10,4))
plt.plot(time_frame)
plt.axvline(fake_voice_start, color='r', linestyle='--', label='fake voice start')
plt.axvline(fake_voice_end, color='r', linestyle='--', label='fake voice finish')
plt.legend()
plt.xlabel('Time Frame')
plt.ylabel('Frequency Change')
plt.show()
```

[사진 4] 변조된 음성이 존재하는 시간대를 찾는 코드

[사진 4]를 통해 변조된 음성의 시작 및 종료 시각을 계산할 수 있다.

```
fake audio start at 38.97747401799117 seconds
fake audio finish at 40.78822027969653 seconds
```

[사진 5] 변조된 음성이 존재하는 부분

이를 통해,

- fake audio start: 38.97747401799117 seconds
- fake audio finish: 40.78822027969653 seconds

임을 알 수 있다.

5. Flag

- fake audio start: 38.97747401799117 seconds
- fake audio finish: 40.78822027969653 seconds

6. 별도 첨부

7. Reference

- <https://sanghyu.tistory.com/37>