

실습 3주차 보고서

00분반 202102675 이문영

과제 2 해결 과정

주어진 WAV 파일을 텍스트로 해석하기

1. WAV 파일을 file2morse 함수를 사용하여 모스 부호로 변환
2. 변환 된 모스부호를 morse_to_text 함수를 사용하여 텍스트로 해석
위와 같이 순차적으로 두 번의 과정으로 WAV파일을 해석합니다.

1. WAV 파일을 모스 부호로 변환

WAVtoMorse.py

전체코드

```
def file2morse(filename):  
    """WAV 파일을 읽어 모스 부호로 변환 (국제 표준 적용)"""  
  
    with wave.open(filename, 'rb') as w:  
        framerate = w.getframerate() # 샘플링 레이트  
(48000Hz)  
        frames = w.getnframes() # 총 프레임 수  
        sampwidth = w.getsampwidth() # 샘플 폭 (바이트 단  
위)  
        audio = []  
  
        for _ in range(frames):  
            frame = w.readframes(1)
```

```

        if sampwidth == 4: # 32비트 PCM (signed)
            audio.append(struct.unpack('<l', frame)
[0])) # 32비트 정수

unit = int(0.1 * framerate) # 100ms 단위 길이
morse = ''
prev_signal = None
silence_count = 0
signal_length = 0

for i in range(0, len(audio), unit):
    segment = audio[i:i+unit] # unit 단위로 신호 분할
    if not segment:
        continue
    stdev = statistics.stdev(segment) # 100ms 단위 신
호의 표준 편차 계산

    if stdev > 10000: # 신호 감지
        signal_length += 1 # 신호 지속 시간 증가
        prev_signal = 'signal'
        silence_count = 0

    else: # 무음 감지
        if prev_signal == 'signal': # 신호 종료 시점
            if 1 <= signal_length <= 2: # 점('.') 감지
(100ms ~ 299ms)

                morse += '.'

            elif signal_length >= 3: # 대시('-') 감지
(300ms 이상)

                morse += '-'
                signal_length = 0 # 신호 길이 초기화

            silence_count += 1
            if silence_count >= 7:
                morse += ' / ' # 단어 구분 (7유닛)
            elif silence_count >= 3:
                morse += ' ' # 문자 구분 (3유닛)

```

```

prev_signal = 'silence'

# 문자 사이 공백을 한 칸으로 변환
morse = morse.replace('  ', ' ').strip()
return morse

```

1. 먼저 읽어들이는 WAV파일의 정보를 저장합니다.

```

with wave.open(filename, 'rb') as w:
    framerate = w.getframerate() # 샘플링 레이트
(48000Hz)
    frames = w.getnframes() # 총 프레임 수
    sampwidth = w.getsampwidth() # 샘플 폭 (바이트 단
위)
    audio = []

```

2. 읽어들이는 WAV 파일의 frame 개수만큼 각 프레임을 32bit PCM으로 언패킹하여 audio 리스트에 저장해줍니다.

```

for _ in range(frames):
    frame = w.readframes(1)
    if sampwidth == 4: # 32비트 PCM (signed)
        audio.append(struct.unpack('<i', frame)[0]) # 32
비트 정수

```

3. unit = 100ms 단위로 신호를 분석합니다. 저장된 audio리스트의 unit 만큼의 segment 리스트를 만들어 segment 즉, unit 단위 만큼의 값들의 표준편차를 계산하고 그 값에 따라 신호가 있는지 혹은 그냥 무음인지 판별합니다.

```

for i in range(0, len(audio), unit):
    segment = audio[i:i+unit] # unit 단위로 신호 분할
    if not segment:
        continue

    stdev = statistics.stdev(segment) # 100ms 단위 신호의 표준 편차 계산

```

4. 각 segment 마다의 표준편차 값을 통해 다음 기준으로 신호를 판별합니다.

- 신호(`stdev > 10000`)가 감지되면
 - `signal_length` 증가 (신호가 지속되는 길이를 측정)
 - `prev_signal = 'signal'` (현재 신호 상태를 기억)
 - `silence_count = 0` (무음 구간 초기화)
- 무음(`stdev <= 10000`)이 감지되면
 - 이전 구간이 신호(`prev_signal == 'signal'`)이었다면 신호 종료 판단
 - 신호 지속 시간이 짧으면 . (점)
 - 신호 지속 시간이 길면 - (대시)
 - 무음 구간(`silence_count`) 증가
 - 문자 간 공백(3유닛)이면 ' ' 추가
 - 단어 간 공백(7유닛)이면 ' / ' 추가
 - `prev_signal = 'silence'` (현재 무음 상태 기록)

```

if stdev > 10000: # 신호 감지
    signal_length += 1 # 신호 지속 시간 증가
    prev_signal = 'signal'
    silence_count = 0

else: # 무음 감지
    if prev_signal == 'signal': # 신호 종료 시점

```

```

        if 1 <= signal_length <= 2: # 점(`.`) 감지
(100ms ~ 299ms)
            morse += '.'
        elif signal_length >= 3: # 대시(`-`) 감지
(300ms 이상)
            morse += '-'
            signal_length = 0 # 신호 길이 초기화

    silence_count += 1
    if silence_count >= 7:
        morse += ' / ' # 단어 구분 (7유닛)
    elif silence_count >= 3:
        morse += ' ' # 문자 구분 (3유닛)

    prev_signal = 'silence'

# 문자 사이 공백을 한 칸으로 변환
morse = morse.replace('  ', ' ').strip()

return morse

```

2. 모스 부호를 텍스트로 반환

morseToText.py

```

import morse_data

# 모스 부호 딕셔너리 가져오기
english = morse_data.english
number = morse_data.number

# 영어 & 숫자 모스 부호를 하나의 딕셔너리로 합치기 (역변환용)
MORSE_DICT = {**english, **number}

```

```

# 기존 딕셔너리를 "모스 부호: 문자" 형태로 변환
REVERSED_MORSE_DICT = {value: key for key, value in
MORSE_DICT.items()}

def morse_to_text(morse_code):
    morse_code = morse_code.strip() # 앞뒤 공백 제거
    words = morse_code.split(' / ') # `/` 기준으로 단어 분
    리
    decoded_text = []

    for word in words:
        # 각 단어 내의 공백을 기준으로 문자를 구분하여 해석 후
        # 하나의 단어로 결합하여 저장
        letters = word.strip().split()
        decoded_word =
''.join(REVERSED_MORSE_DICT.get(letter, '?') for letter in
letters)

        decoded_text.append(decoded_word)

    return ' '.join(decoded_text) # 단어 간 공백 추가

```

1. 제공된 morse_data의 모스부호 딕셔너리를 가져와 하나의 딕셔너리로 만들어 역변환에 용이하게 가공합니다.
2. 이를 기반으로 morse_to_text 함수를 작성하였습니다.
 1. 혹시 있을 앞 뒤 공백을 제거해주고, 단어간의 구분자인 '/'를 기준으로 단어를 분리해줍니다.

2. 분리된 단어들의 리스트 **words**를 반복문으로 각 단어 내의 공백을 기준으로 **문자**를 구분하여 **letters**에 저장합니다.
3. 각 **letter**에 대하여 REVERSED_MORSE_DICT.get()을 활용하여 현재 letter가 딕셔너리에 대응하는 값이 있는지 탐색하고, 만약 그 값이 존재하면 decoded_word에 공백없이 붙여서 저장해줍니다.
4. 이렇게 각 단어마다 각 문자들을 해석하고 해석된 단어를 decoded_text 리스트에 추가해줍니다.
5. 마지막으로 decoded_text 리스트의 원소들을 각 원소 사이에 공백을 추가하여 문자열 형태로 반환해줍니다.

3. 최종 WAVtoText.py 및 실행 결과

```
from WAVtoMorse import file2morse # WAV → Morse 변환 코드
from morseToText import morse_to_text # Morse → Text 변환 코드
```

```
def wav_to_text(filename):
```

```
    # **Step 1: WAV → Morse 변환**
```

```
    morse_code = file2morse(filename)
```

```
    print(f"Extracted Morse Code: {morse_code}")
```

```
    # **Step 2: Morse → Text 변환**
```

```
    decoded_text = morse_to_text(morse_code)
```

```
    print(f"Decoded Text: {decoded_text}")
```

```
    return decoded_text
```

```
wav_filename = "wavs/output_202102675_이문영.wav" # 변환할 WAV 파일 이름
```

```
text_result = wav_to_text(wav_filename)

print(f"최종 변환 결과: {text_result}")

txt_filename = "202102675_이문영.txt"

with open(txt_filename, "w", encoding="utf-8") as txt_file:
    txt_file.write(text_result)

print(f"변환된 텍스트가 파일로 저장되었습니다: {txt_filename}")
```

저에게 주어진 WAV파일은 48000Hz의 샘플링 레이트, 2486400의 프레임 수, 32비트의 샘플 폭을 가졌으며 이를 모스 부호로 해독한 결과는

이고, 이를 텍스트로 변환한 결과는

과제 1 해결과정

1. 입력받은 텍스트를 모스부호로 변환

textToMorse.py

```
import morse_data

english = morse_data.english
number = morse_data.number

def text2morse(text):
    text = text.upper() # 대문자로 변환
    words = text.split() # 단어 단위로 분리
    morse_words = []

    for word in words:
        morse_chars = []
        for t in word:
            if t in english:
                morse_chars.append(english[t])
            elif t in number:
                morse_chars.append(number[t])
        morse_words.append(' '.join(morse_chars)) # 문자
        # 사이 한 칸 공백 추가

    return '/ '.join(morse_words) # 단어 사이 한 칸 공백
    # 추가
```

1. 입력 받은 텍스트를 단어 단위로 분리하여 words 리스트에 저장합니다.
2. 각 word의 문자 하나 하나를 모스부호 딕셔너리에 존재하는지 판별 후 그 값을 morse_chars에 저장합니다.
3. 각 모스부호 문자 사이에 공백으로 구분 짓고, 단어 사이는 /로 구분 짓습니다.

2. 변환된 모스부호를 WAV파일로 변환

과제 설명란에 명시된 WAV 파일 형식을 지켜 WAV 파일을 생성합니다.

MorseToWAV.py

```
import wave
import struct
import math

# 기본 설정
INTMAX = 2**((32-1)-1) # 32-bit PCM 최대값 (2147483647)
UNIT_TIME = 0.1 # 모스 부호의 기본 단위 (100ms)
FS = 48000 # 샘플링 레이트 (48kHz)
FREQ = 523.251 # 기본 주파수 (C5)

def morse2audio(morse):
    """모스 부호를 오디오 샘플 데이터로 변환 (국제 표준 적용)"""
    audio = []

    for symbol in morse:
        if symbol == '.': # dit (점) - 1유닛
            audio.extend(generate_tone(UNIT_TIME * 1))
        elif symbol == '-': # dah (대시) - 3유닛
            audio.extend(generate_tone(UNIT_TIME * 3))
        elif symbol == ' ': # 문자 간 공백 - 3유닛
            audio.extend(generate_silence(UNIT_TIME * 3))
        elif symbol == '/': # 단어 간 공백 - 7유닛
            audio.extend(generate_silence(UNIT_TIME * 7))

    # 점과 대시 사이 간격 (1유닛 공백 추가)
    audio.extend(generate_silence(UNIT_TIME * 1))

    return audio
```

```

def generate_tone(duration):
    """ 특정 주파수(FREQ)의 사인파를 duration(초) 동안 생성 """
    samples = int(FS * duration)
    return [int(INTMAX * math.sin(2 * math.pi * FREQ * (i / FS))) for i in range(samples)]

def generate_silence(duration):
    """ duration(초) 동안 무음(0) 추가 """
    samples = int(FS * duration)
    return [0] * samples

def audio2file(audio, filename):
    """ 오디오 데이터를 32비트 PCM WAV 파일로 저장 """
    with wave.open(filename, 'wb') as w:
        w.setnchannels(1) # 모노 채널
        w.setsampwidth(4) # 32-bit PCM (4바이트)
        w.setframerate(FS) # 샘플링 레이트 48kHz

        for a in audio:
            w.writeframes(struct.pack('<l', a)) # 32-bit
정수 저장

```

3. textToWAV.py 및 결과

```

from textToMorse import text2morse # Text → Morse 변환 코드
from MorseToWAV import morse2audio, audio2file # Morse → WAV 변환 코드

def text_to_wav(text, filename):
    """ 텍스트를 모스 부호로 변환 후 WAV 파일로 저장 """

    # **Step 1: Text → Morse 변환**


```


```
morse_code = text2morse(text)
print(f"Converted Morse Code: {morse_code}")

# **Step 2: Morse → WAV 변환**
audio_data = morse2audio(morse_code)

# **Step 3: WAV 파일 저장**
audio2file(audio_data, filename)
print(f"WAV 파일이 생성되었습니다: {filename}")
```

Or analyse an audio file containing Morse code:

Upload 

Play 

Stop 

Filename: "202102675_이문영.wav"

CNU CSE 202102675 LEEMOONYOUNG