Speech Processing Assignment - 2

GitHub Link : https://github.com/Toji339/SpeechProcessing

Dataset Description :

- This LJ Speech dataset has been taken from the Kaggle.
- It contains 13,100 short audio clips of a single speaker
- These clips vary in length from 1 to 10 sec and have a total length of around 24 hours.

Objective :

-
  - Load and preprocess a speech signal (convert to mono and resample to 16kHz).
  - Use a pre-trained Wav2Vec2 model to recognize phonemes from the speech signal.
  - Extract specific phoneme segments based on their time intervals.
  - Visualize the phoneme waveforms and label them according to the recognized phonemes.

Code :

```python

import torchaudio
import torchaudio.transforms as T
import librosa.display
import torch
from transformers import Wav2Vec2Processor, Wav2Vec2ForCTC
import matplotlib.pyplot as plt
import numpy as np
import nltk
from nltk.corpus import cmudict

nltk.download('cmudict')
cmu_dict = cmudict.dict()

audio_path = "/content/LJ001-0030.wav"
waveform, sample_rate = torchaudio.load(audio_path)

resampler = T.Resample(orig_freq=sample_rate, new_freq=16000)
waveform = resampler(waveform.mean(dim=0, keepdim=True))

processor = Wav2Vec2Processor.from_pretrained("facebook/wav2vec2-large-960h")
model = Wav2Vec2ForCTC.from_pretrained("facebook/wav2vec2-large-960h")

```

```python
input_values = processor(waveform.squeeze().numpy(), return_tensors="pt", sampling_rate=16000).input_values
with torch.no_grad():
    logits = model(input_values).logits


predicted_ids = torch.argmax(logits, dim=-1)
transcription = processor.batch_decode(predicted_ids)[0].upper()
print("Recognized Text:", transcription)

words = transcription.split()
phonemes = []
for word in words:
    if word in cmu_dict:
        phonemes.extend(cmu_dict[word][0])
    else:
        phonemes.append(word)

print("Recognized Phonemes:", " ".join(phonemes))

total_duration = waveform.shape[1] / 16000
time_intervals = np.linspace(0, total_duration, num=len(phonemes))

plt.figure(figsize=(12, 4))
librosa.display.waveshow(waveform.squeeze().numpy(), sr=16000)
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.title("Phoneme Visualization")


for i, phoneme in enumerate(phonemes):
    if i < len(time_intervals):
        plt.text(time_intervals[i], 0, phoneme, fontsize=10, ha='center', color='red', rotation=45)

plt.show()
```

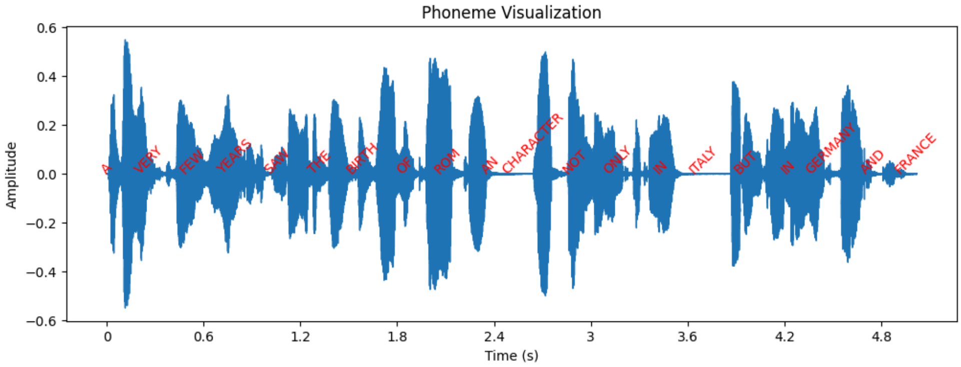

Figure 1: Final Plot with recognized text .