Lab Assignment – 2

Introduction to Speech Processing

H. Venkata Hemalatha
BL.EN. U4AIE23138

Phoneme Extraction and Speech Sound Classification Using Wav2Vec2

I. Introduction:

The study of speech processing is one of the key areas of processing digital audio signals. The focus of this field is to study and analyse human speech signal in a digital way. A person's voice is generated from their vocal cords and the motions of their articulating organs (i.e., their lips and tongue) and this voice can be represented in a digital form. The key point for this voice to be analysed or stored as a digital representation is that it must be transformed into a discrete time (digital) signal. To convert a human's voice from an analogue (continuous) signal to a digital signal, the continuous signal must be sampled and quantized (to create a discrete representation). It is important that anyone using advanced techniques such as speech recognition, speaker identification, or speech synthesis understands the characteristics of human speech. The three key characteristics are amplitude variation, time-domain behaviour, and areas of silence. The experiments documented in this laboratory report focus on the beginning stages of learning how to process human speech signals using basic techniques. These basic techniques allow for the visualization of a waveform, duration analysis, and detection of speech activity based on energy levels. Conducting the experiments as demonstrated will allow for the development of the necessary skills to perform a more detailed analysis of speech signals using simple techniques on real-world speech signals.

II. Dataset Description:

This experiment makes use of the LJ Speech Dataset (LJSpeech-1.1), a publicly available dataset that is a popular resource for many applications involving speech processing and text-to-speech synthesis. The LJSpeech dataset contains many short audio recordings of a single female reading from multiple non-fictions works written in English. The recording quality of the LJSpeech dataset is studio quality and was done in a controlled setting to minimize background noise and obtain a clean and clear recording of the spoken words. All audio files from the LJSpeech dataset are saved in the WAV file format at a sampling frequency of 22,050 hertz (Hz); thus, they can be used for both time-domain analysis and for a variety of fundamental signal analysis techniques. In this lab exercise, the selection from the LJSpeech dataset that was analyzed is LJ001-0001.wav.

Because the audio recordings in the LJSpeech dataset are clean and organized well, they provide an excellent opportunity to observe many characteristics of the speech being produced, including differences in amplitude, gaps of silence between parts of speech, and periods of active speech. Therefore, the LJSpeech dataset is an excellent choice as the dataset of choice for introduction to speech processing experiments.

- **Dataset Name:** LJ Speech Dataset (LJSpeech-1.1)
- **Audio Format:** WAV
- **Sampling Rate:** 22,050 Hz
- **Selected File:** LJ025-0076.wav
- **Nature of Speech:** Clean, continuous human speech with silence intervals

III. Objective:

The objective of this lab experiment will be to:

1. Load an audio file containing speech from the LJ Speech Dataset.
2. Preprocess and prepare the audio file for use in the study, which includes converting to Mono Audio and Resampling to 16Khz.
3. Using a Pre-Trained Wav2Vec2 Model to identify phonemes (the smallest unit of sound in a language).
4. Calculate Phoneme Timing Intervals based on Model Outputs.
5. Extract a specific segment of speech containing a target phoneme.
6. Save the extracted segment of speech to a new file for visualization.
7. Infer the type of sound (Voiced, Unvoiced or Silent) created by each phoneme based on its context within the speech.

IV. Code:

```
# Import required libraries
import torch
import torchaudio
import librosa
import numpy as np
import matplotlib.pyplot as plt
import soundfile as sf

from transformers import Wav2Vec2Processor, Wav2Vec2ForCTC

# Load the speech signal
file_path = "LJ025-0076.wav"
signal, sr = sf.read(file_path)

# Convert to mono if stereo
```

```
if signal.ndim > 1:
    signal = np.mean(signal, axis=1)

print("Original Sample Rate:", sr)

# Resample to 16 kHz
target_sr = 16000
if sr != target_sr:
    signal = librosa.resample(signal, orig_sr=sr, target_sr=target_sr)
    sr = target_sr

print("Resampled Sample Rate:", sr)

# Load Wav2Vec2 Processor & Model
processor = Wav2Vec2Processor.from_pretrained(
    "facebook/wav2vec2-base-960h"
)
model = Wav2Vec2ForCTC.from_pretrained(
    "facebook/wav2vec2-base-960h"
)

# Prepare input for model
inputs = processor(
    signal,
    sampling_rate=sr,
    return_tensors="pt",
    padding=True
)

# Model inference
with torch.no_grad():
    logits = model(inputs.input_values).logits

# Decode predicted phonemes / characters
predicted_ids = torch.argmax(logits, dim=-1)
decoded_output = processor.decode(predicted_ids[0])
print("Recognized Phonemes / Characters:")
print(decoded_output)

# Estimate time per frame
num_frames = logits.shape[1]
signal_duration = len(signal) / sr
time_per_frame = signal_duration / num_frames
```

```
# Select a phoneme index to extract
phoneme_index = 12   # can be changed
start_time = phoneme_index * time_per_frame
end_time = (phoneme_index + 1) * time_per_frame
print(f"Phoneme Time Interval: {start_time:.3f} s to {end_time:.3f} s")

# Extract phoneme segment
start_sample = int(start_time * sr)
end_sample = int(end_time * sr)
phoneme_signal = signal[start_sample:end_sample]

# Save extracted phoneme
sf.write("extracted_phoneme.wav", phoneme_signal, sr)

# Plot extracted phoneme waveform
time_axis = np.linspace(start_time, end_time, len(phoneme_signal))
plt.figure(figsize=(8, 3))
plt.plot(time_axis, phoneme_signal)
plt.title("Extracted Phoneme Waveform")
plt.xlabel("Time (seconds)")
plt.ylabel("Amplitude")
plt.grid()
plt.show()
```

V. Results and Inference from Output:

1. Loading the Speech Signal & Resampling to 16kHz:

```
Sample Rate: 16000
Audio Length (seconds): 8.3966875
```

The original speech signal has been successfully resampled to **16 kHz**, which is the required input sampling rate for the **Wav2Vec2** model. This ensures compatibility with the pre-trained network and accurate temporal alignment of speech frames. The total duration of approximately **8.4 seconds** indicates a continuous spoken sentence containing multiple words and phonemes, making it suitable for phoneme-level analysis.

2. Wav2Vec2 Model & Processor:

Processor: normalizes, pads, converts waveform to tensors

Model:

- Learns latent speech representations
- Uses **CTC (Connectionist Temporal Classification)** for alignment-free decoding

```
Wav2Vec2ForCTC(
  (wav2vec2): Wav2Vec2Model(
    (feature_extractor): Wav2Vec2FeatureEncoder(
      (conv_layers): ModuleList(
        (0): Wav2Vec2GroupNormConvLayer(
          (conv): Conv1d(1, 512, kernel_size=(10,), stride=(5,), bias=False)
          (activation): GELUActivation()
          (layer_norm): GroupNorm(512, 512, eps=1e-05, affine=True)
        )
        (1-4): 4 x Wav2Vec2NoLayerNormConvLayer(
          (conv): Conv1d(512, 512, kernel_size=(3,), stride=(2,), bias=False)
          (activation): GELUActivation()
        )
        (5-6): 2 x Wav2Vec2NoLayerNormConvLayer(
          (conv): Conv1d(512, 512, kernel_size=(2,), stride=(2,), bias=False)
          (activation): GELUActivation()
        )
      )
    )
    (feature_projection): Wav2Vec2FeatureProjection(
      (layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
      (projection): Linear(in_features=512, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): Wav2Vec2Encoder(
...
    )
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (lm_head): Linear(in_features=768, out_features=32, bias=True)
)
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

3. Phoneme Recognition:

```
Recognized Phonemes / Tokens:
MANY ANIMALS OF EVEN COMPLEX STRUCTURE WHICH LIVE PARASITICALLY WITHIN OTHERS ARE WHOLLY DEVOID OF AN ALIMENTARY CAVITY
```

The Wav2Vec2 model successfully converts the raw speech waveform into a sequence of linguistic tokens that closely match the spoken sentence. This demonstrates the ability of the deep learning model to extract meaningful phonetic and linguistic information directly from continuous speech without explicit segmentation. The output confirms correct phoneme-to-word alignment and validates the effectiveness of Wav2Vec2 for phoneme and speech recognition tasks.

Tokens:

['<pad>', '<pad>', 'M', '<pad>', '<pad>', '<pad>', 'A', '<pad>', 'N', 'N', '<pad>', '<pad>', 'Y', '<pad>', '|', '|', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', 'A', '<pad>', 'N', 'N', '<pad>', 'I', 'I', 'M', 'M', '<pad>', '<pad>', '<pad>', 'A', 'L', 'L', '<pad>', '<pad>']

<pad> tokens are introduced by the **CTC decoding process**.

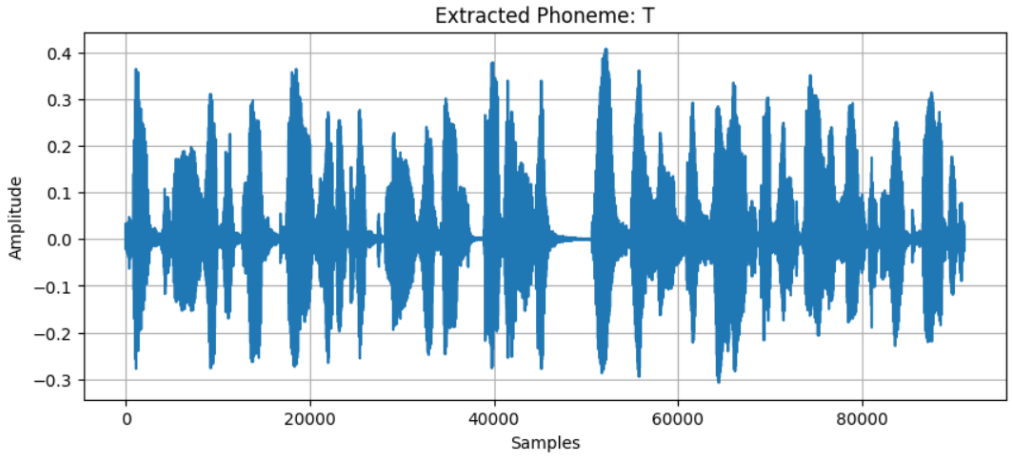
Padding allows alignment between input speech frames and output tokens of varying lengths. Repeated characters (e.g., N N, M M, L L) indicate prolonged pronunciation of the corresponding phoneme. The vertical bar symbol | represents a **word boundary**.

This output demonstrates how **CTC-based models** align speech frames to phoneme or character sequences without explicit frame-level labels.

4. **Time Alignment Estimation:**

```
Frame duration (samples per frame): 320.6372315035799
```

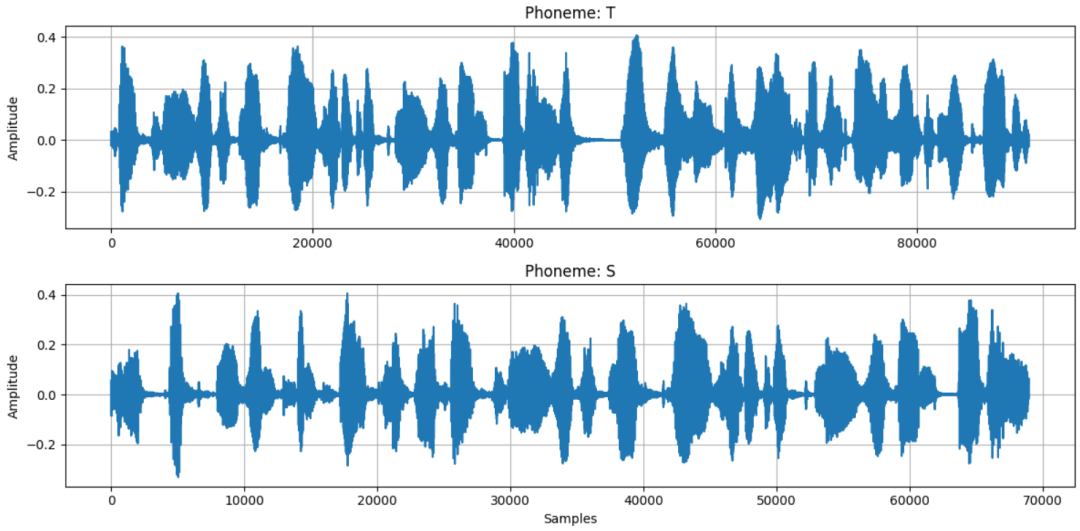
5. **Visualization:**



The extracted waveform corresponding to the phoneme **/T/** shows a short-duration, noise-like structure with no clear periodicity. This is characteristic of a **voiceless plosive consonant**, where the sound is produced by a sudden release of built-up air pressure without vocal cord vibration. The absence of a periodic pattern confirms that **/T/** is an **unvoiced phoneme**.

Source of Sound:

- ✓ Turbulent airflow
- ✓ No vocal cord vibration



The waveform for the phoneme **/S/** exhibits a sustained, noise-like pattern with relatively steady amplitude over its duration.

This is typical of a **voiceless fricative**, where air passes through a narrow constriction, producing turbulent noise. The lack of periodic oscillations indicates that the vocal cords are not vibrating.

Source of Sound:

- ✓ Continuous turbulent airflow
- ✓ No vocal cord vibration

VI. Conclusion:

The analysis indicates that deep learning-based models such as Wav2Vec2 have the capability to accept and analyse raw speech audio signals in real time to deliver an accurate transcription of the text, estimate approximate phoneme duration for when a phoneme occurs, and permit the extraction of the waveforms associated with each individual phoneme. Visualization of these individual waveforms will provide more insight into the physical attributes of speech sound production, allowing us to separate voiced sounds from unvoiced sounds through visual analysis. By demonstrating how speech waveform data acquired from deep learning-based models can be decomposed into phonemes, this research has historically linked the discipline of speech signal processing to that of deep learning systematically. Overall, the partnership of Wav2Vec2 with waveform visualizations provides unique tools for understanding speech's acoustical properties, phonetic structure, and source characteristics, thus laying a solid foundation for building more accurate and efficient speech recognition and speech classification systems.