# Lab 3 - Speech Processing

**Sanjushree Rajan**

**BL.EN.U4AIE23130**

## Objective 1

To identify, segment, and visualize different phoneme classes in a recorded speech signal and to analyze voiced and unvoiced regions using waveform-level inspection.

Record the sentence: "*She judged the blue pen, put it on the table, and quietly whispered yes.*"

Using the procedures implemented in Lab-2, perform the following steps:

1. Load and preprocess the recorded speech signal.
2. Automatically segment the speech into phoneme-level units (similar to previous lab).
3. Plot the time-domain waveform corresponding to each extracted phoneme.
4. For every phoneme segment:
   - Identify its phoneme category (vowel, plosive, fricative, affricate, semivowel, diphthong, or whisper segment).
   - Label the phoneme accordingly.
5. Select and extract:
   - One voiced phoneme segment
   - One unvoiced phoneme segment
6. Compare the waveform characteristics of the voiced and unvoiced segments and explain the observed differences based on:
   - Periodicity
   - Amplitude patterns
   - Presence or absence of noise-like structure

## 1. Library Imports

In [1]:
```
import torch
import torchaudio
import numpy as np
import matplotlib.pyplot as plt
import librosa

from transformers import Wav2Vec2Processor, Wav2Vec2ForCTC

import warnings
warnings.filterwarnings("ignore")
```

```
C:\Users\Srijjay\AppData\Roaming\Python\Python313\site-packages\tqdm\auto.py:21:
TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See http
s://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
```

# 2. Load Pre-trained Wav2Vec2 Model

In [2]:
```
processor = Wav2Vec2Processor.from_pretrained("facebook/wav2vec2-base-960h")
model = Wav2Vec2ForCTC.from_pretrained("facebook/wav2vec2-base-960h")
model.eval()
```

```
Loading weights: 100%|██████████| 212/212 [00:01<00:00, 184.18it/s, Materializing
param=wav2vec2.feature_projection.projection.weight]
Wav2Vec2ForCTC LOAD REPORT from: facebook/wav2vec2-base-960h
Key                          | Status  |
-----------------------------+---------+-
wav2vec2.masked_spec_embed | MISSING |

Notes:
- MISSING       :those params were newly initialized because missing from the che
ckpoint. Consider training on your downstream task.
```

Out[2]:
```
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(
      (pos_conv_embed): Wav2Vec2PositionalConvEmbedding(
        (conv): ParametrizedConv1d(
          768, 768, kernel_size=(128,), stride=(1,), padding=(64,), groups=16
          (parametrizations): ModuleDict(
            (weight): ParametrizationList(
              (0): _WeightNorm()
            )
          )
        )
        (padding): Wav2Vec2SamePadLayer()
        (activation): GELUActivation()
      )
      (layer_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
      (layers): ModuleList(
        (0-11): 12 x Wav2Vec2EncoderLayer(
          (attention): Wav2Vec2Attention(
            (k_proj): Linear(in_features=768, out_features=768, bias=True)
            (v_proj): Linear(in_features=768, out_features=768, bias=True)
            (q_proj): Linear(in_features=768, out_features=768, bias=True)
            (out_proj): Linear(in_features=768, out_features=768, bias=True)
          )
          (dropout): Dropout(p=0.1, inplace=False)
          (layer_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=True)
          (feed_forward): Wav2Vec2FeedForward(
            (intermediate_dropout): Dropout(p=0.1, inplace=False)
            (intermediate_dense): Linear(in_features=768, out_features=3072, bi
as=True)
            (intermediate_act_fn): GELUActivation()
            (output_dense): Linear(in_features=3072, out_features=768, bias=Tru
e)
            (output_dropout): Dropout(p=0.1, inplace=False)
          )
          (final_layer_norm): LayerNorm((768,), eps=1e-05, elementwise_affine=T
rue)
```

```
        )
      )
    )
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (lm_head): Linear(in_features=768, out_features=32, bias=True)
)
```

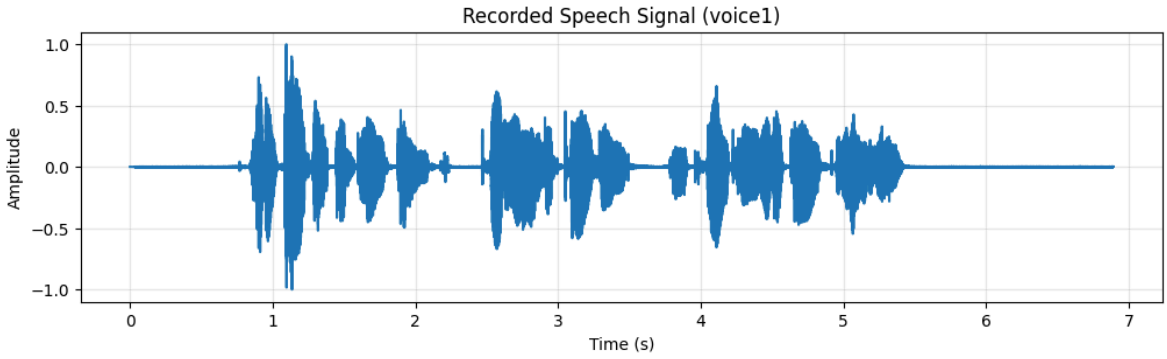
# 3. Load and Preprocess Speech Signal

In [3]:
```
import os
import librosa
import torch

audio_filename = "sentence1.wav"


if not os.path.exists(audio_filename):
    print(f"Error: '{audio_filename}' not found. Please upload your recorded aud
else:
    data, sample_rate = librosa.load(audio_filename, sr=None, mono=False)

    waveform = torch.from_numpy(data).float()

    if len(waveform.shape) == 1:
        waveform = waveform.unsqueeze(0)

    if waveform.shape[0] > 1:
        waveform = torch.mean(waveform, dim=0, keepdim=True)

    waveform = waveform / torch.max(torch.abs(waveform))

    print("Sampling Rate:", sample_rate)
```

```
Sampling Rate: 48000
```

# 4. Plot Original Speech Waveform

In [4]:
```
time_axis = np.arange(waveform.shape[1]) / sample_rate

plt.figure(figsize=(12,3))
plt.plot(time_axis, waveform[0].numpy())
plt.title("Recorded Speech Signal (voice1)")
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.grid(alpha=0.3)
plt.show()
```




# 5. Recognize Phonemes using Wav2Vec2

In [5]:
```
target_sampling_rate = 16000

if sample_rate != target_sampling_rate:
    waveform_np = waveform[0].numpy()
    waveform_resampled = librosa.resample(waveform_np, orig_sr=sample_rate, targ
    input_sr = target_sampling_rate
else:
    waveform_resampled = waveform[0].numpy()
    input_sr = sample_rate

input_values = processor(
    waveform_resampled,
    sampling_rate=input_sr,
    return_tensors="pt"
).input_values

with torch.no_grad():
    logits = model(input_values).logits

predicted_ids = torch.argmax(logits, dim=-1)
transcription = processor.decode(predicted_ids[0])

print("Recognized Phoneme Sequence:")
print(transcription)
```

```
Recognized Phoneme Sequence:
SHE JUDGED THE BLUE PEN PUT IT ON THE TABLE AND QUIETLY WHISPERED YES
```

# 6. Estimate Phoneme Time Intervals

In [6]:
```
predicted_tokens = predicted_ids[0].numpy()
time_step = waveform.shape[1] / logits.shape[1] / sample_rate

phoneme_intervals = []
current_phoneme = None
start_idx = 0

for idx, token_id in enumerate(predicted_tokens):
    phoneme = processor.decode([token_id]).strip()

    if phoneme and phoneme != current_phoneme:
        if current_phoneme is not None:
            phoneme_intervals.append({
```

```
                "phoneme": current_phoneme,
                "start_sample": int(start_idx * time_step * sample_rate),
                "end_sample": int(idx * time_step * sample_rate)
            })
        current_phoneme = phoneme
        start_idx = idx

if current_phoneme is not None:
    phoneme_intervals.append({
        "phoneme": current_phoneme,
        "start_sample": int(start_idx * time_step * sample_rate),
        "end_sample": waveform.shape[1]
    })

print(f"Detected {len(phoneme_intervals)} phoneme segments")
```

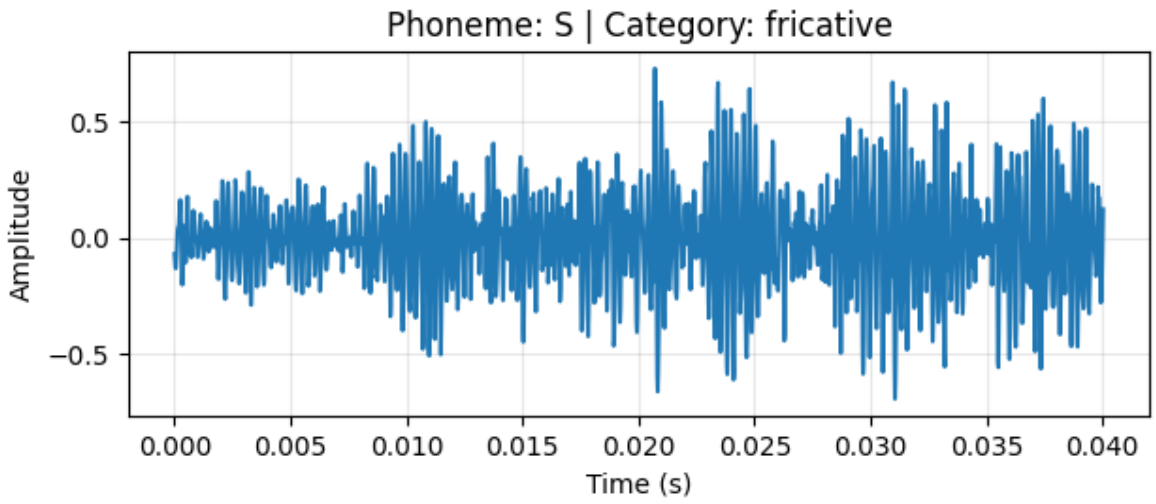
```
Detected 56 phoneme segments
```

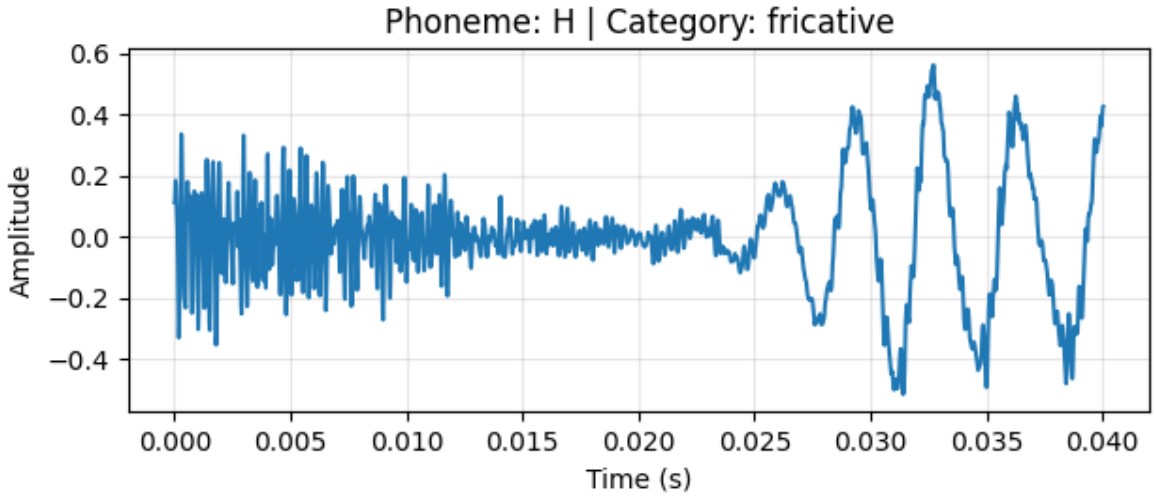
# 7. Phoneme Category Classification

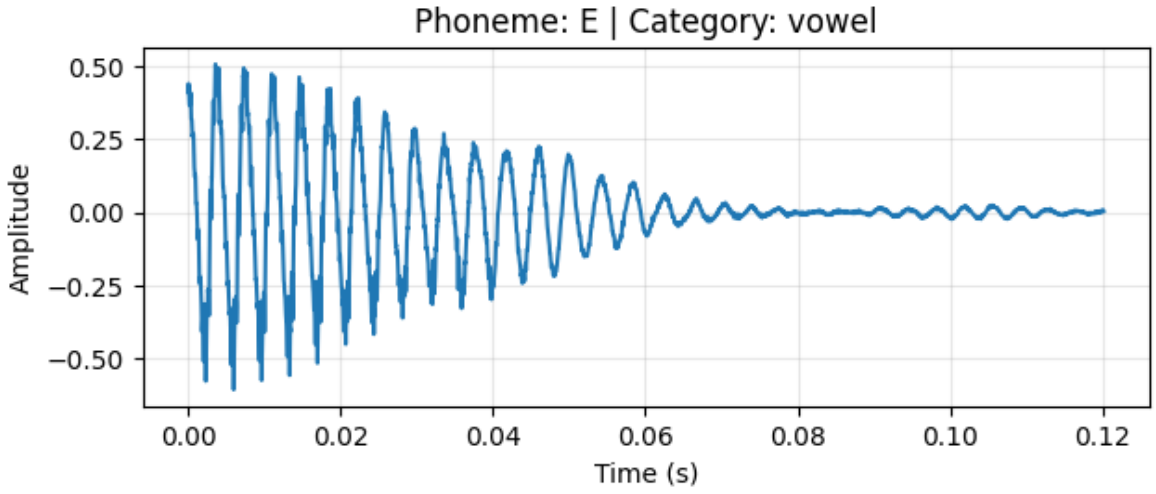
In [7]:
```
phoneme_categories = {
    'A':'vowel', 'E':'vowel', 'I':'vowel', 'O':'vowel', 'U':'vowel',
    'P':'plosive', 'B':'plosive', 'T':'plosive', 'D':'plosive', 'K':'plosive', '
    'F':'fricative', 'S':'fricative', 'H':'fricative', 'Z':'fricative',
    'Y':'semivowel', 'W':'semivowel', 'L':'semivowel', 'R':'semivowel'
}

for p in phoneme_intervals:
    p["category"] = phoneme_categories.get(p["phoneme"].upper(), "whisper/other"
```

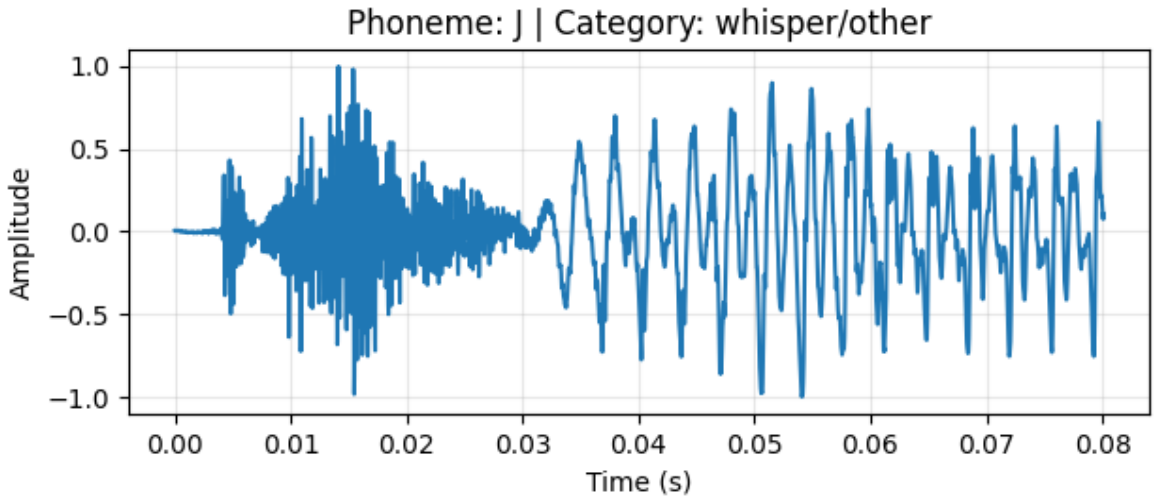
# 8. Extract & Plot Each Phoneme Waveform

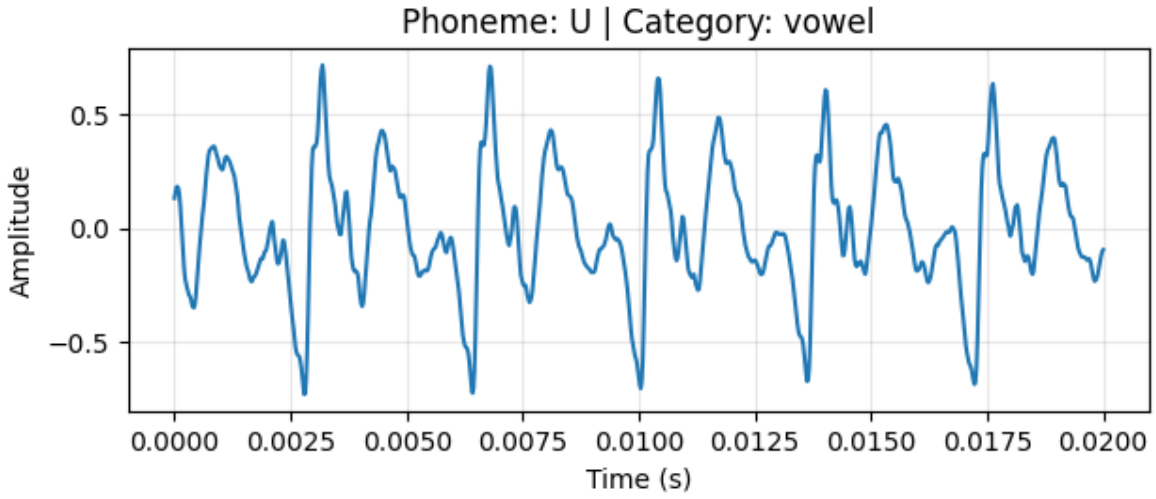
In [8]:
```
def plot_phoneme(segment, sr, phoneme, category):
    t = np.arange(len(segment)) / sr
    plt.figure(figsize=(7,2.5))
    plt.plot(t, segment)
    plt.title(f"Phoneme: {phoneme} | Category: {category}")
    plt.xlabel("Time (s)")
    plt.ylabel("Amplitude")
    plt.grid(alpha=0.3)
    plt.show()

for p in phoneme_intervals:
    segment = waveform[0, p["start_sample"]:p["end_sample"]].numpy()
    plot_phoneme(segment, sample_rate, p["phoneme"], p["category"])
```

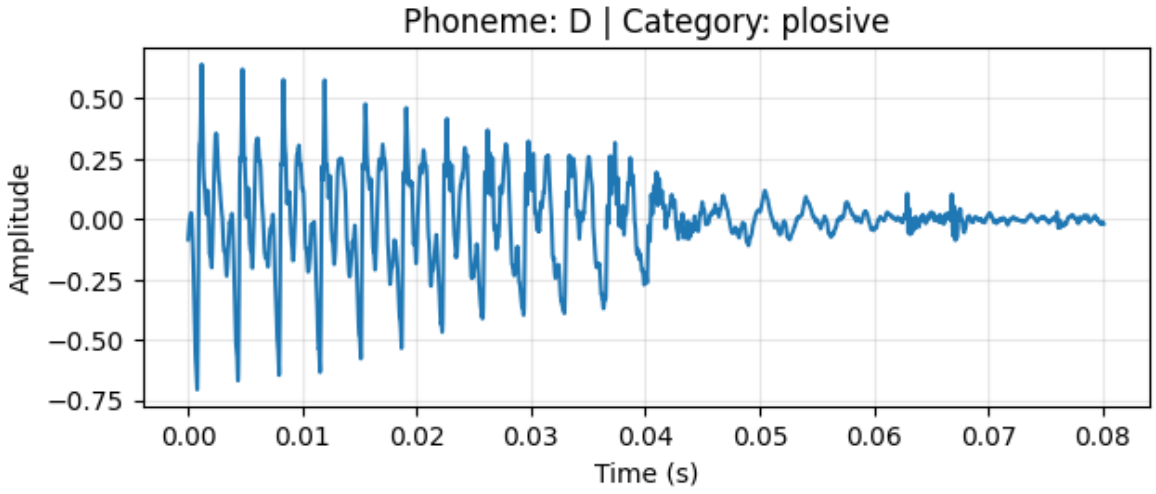

Phoneme: S | Category: fricative
Amplitude
0.5
0.0
−0.5
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
Time (s)

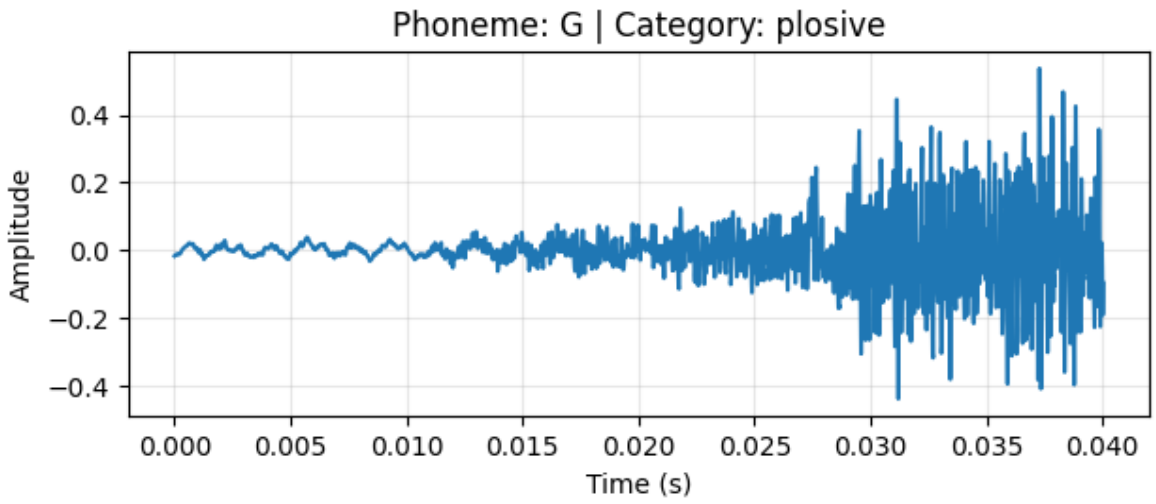

Phoneme: H | Category: fricative
Amplitude
0.6
0.4
0.2
0.0
−0.2
−0.4
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
Time (s)

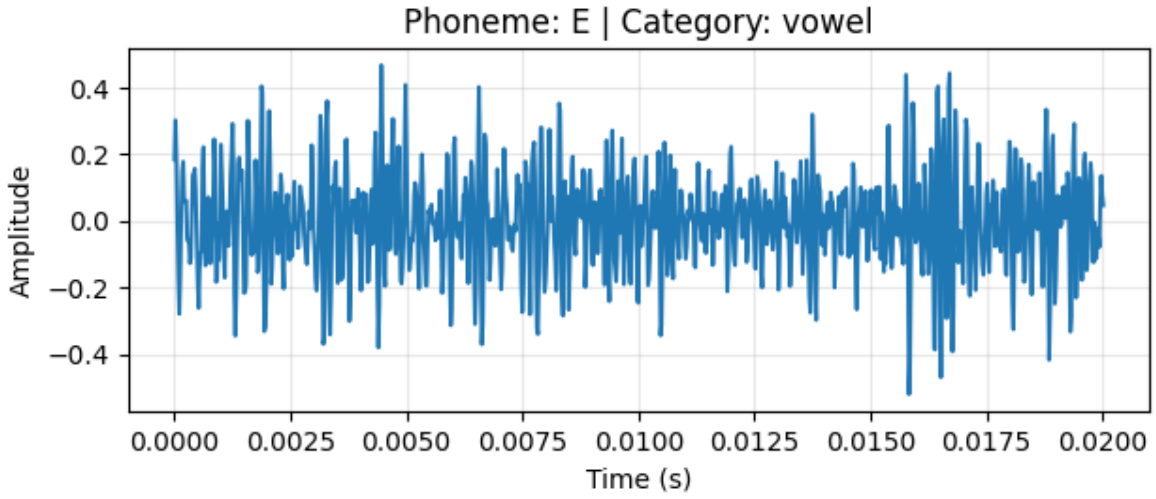

Phoneme: E | Category: vowel
Amplitude
0.50
0.25
0.00
−0.25
−0.50
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Time (s)

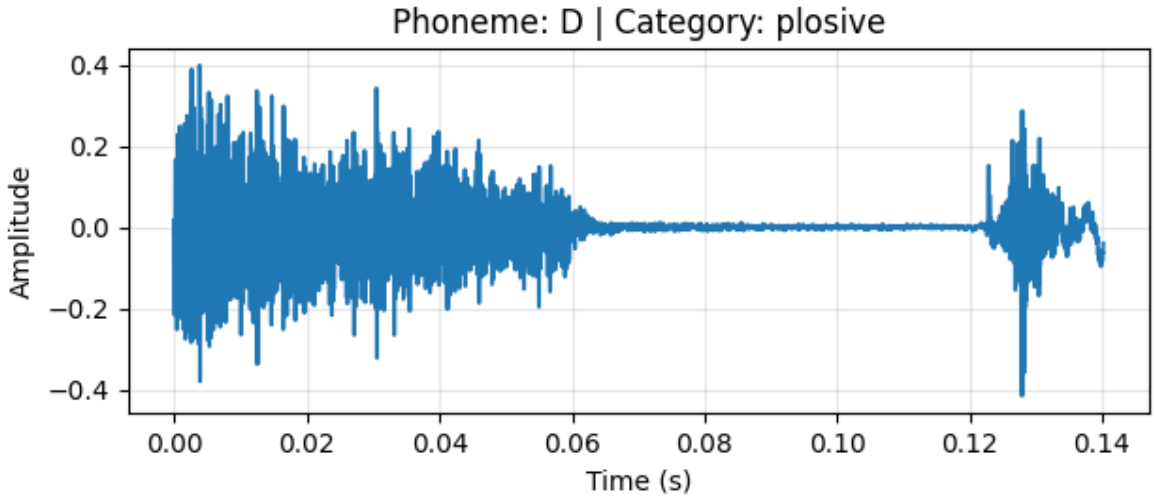

Phoneme: J | Category: whisper/other
Amplitude
Time (s)

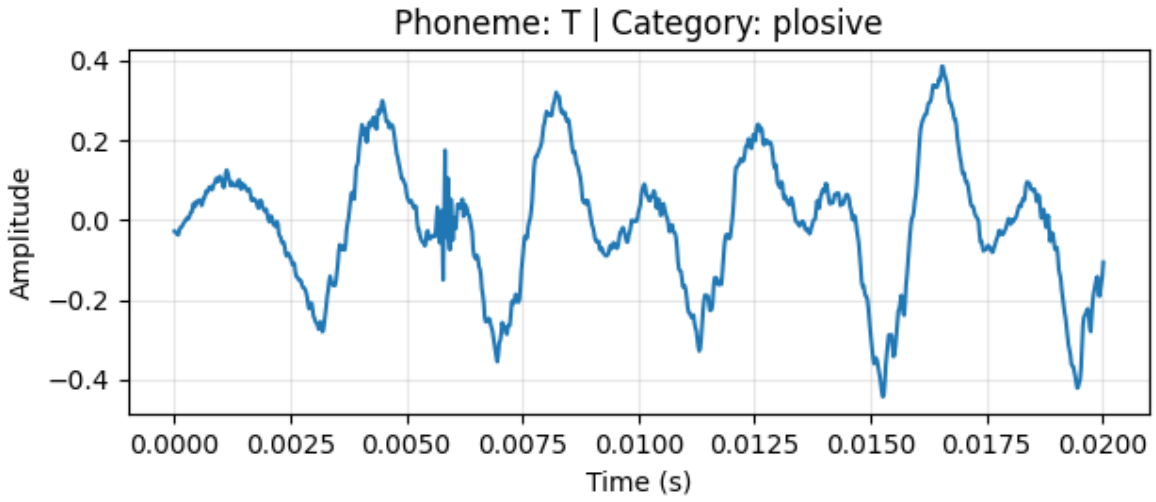

Phoneme: U | Category: vowel
Amplitude
Time (s)

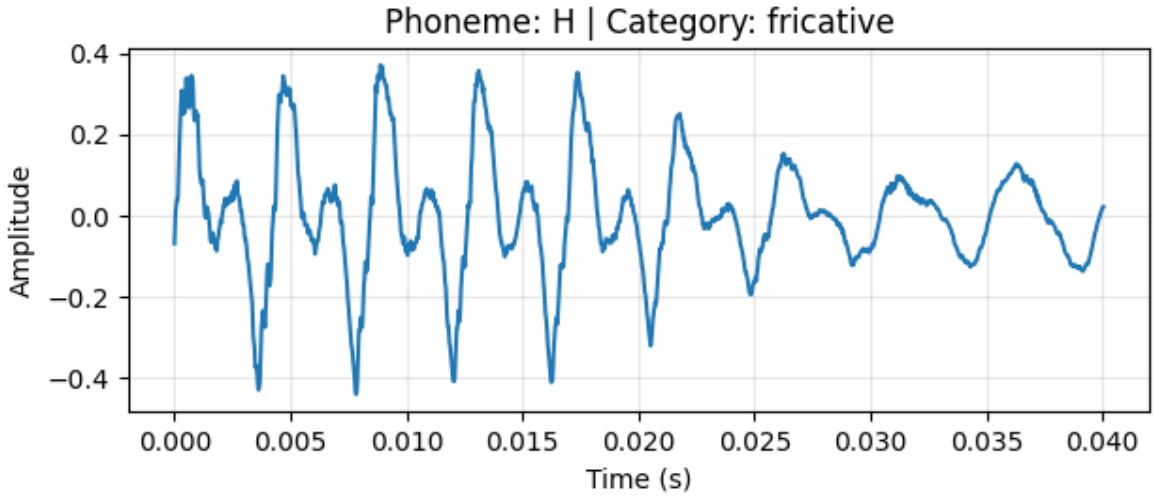

Phoneme: D | Category: plosive
Amplitude
Time (s)

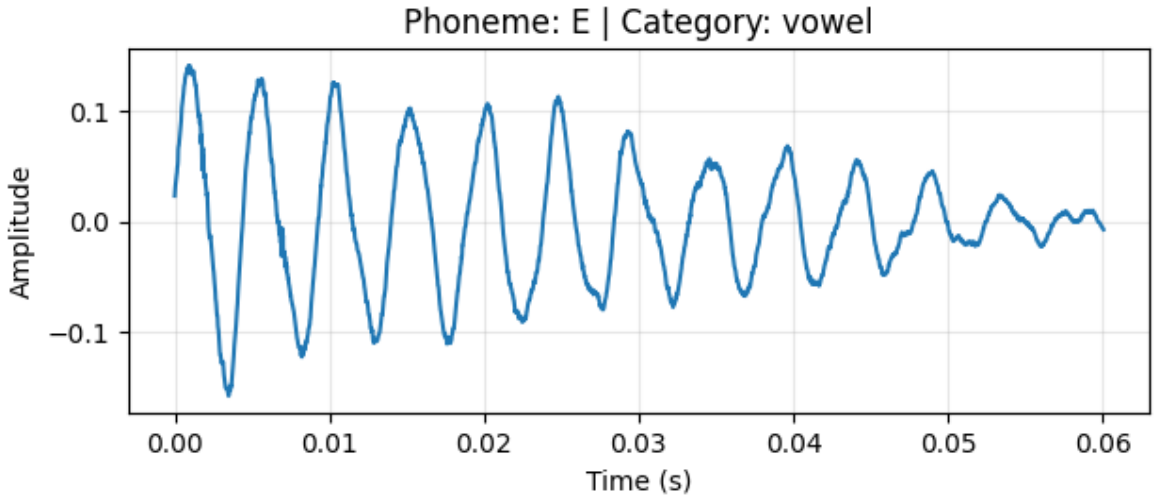

Phoneme: G | Category: plosive
Amplitude
Time (s)

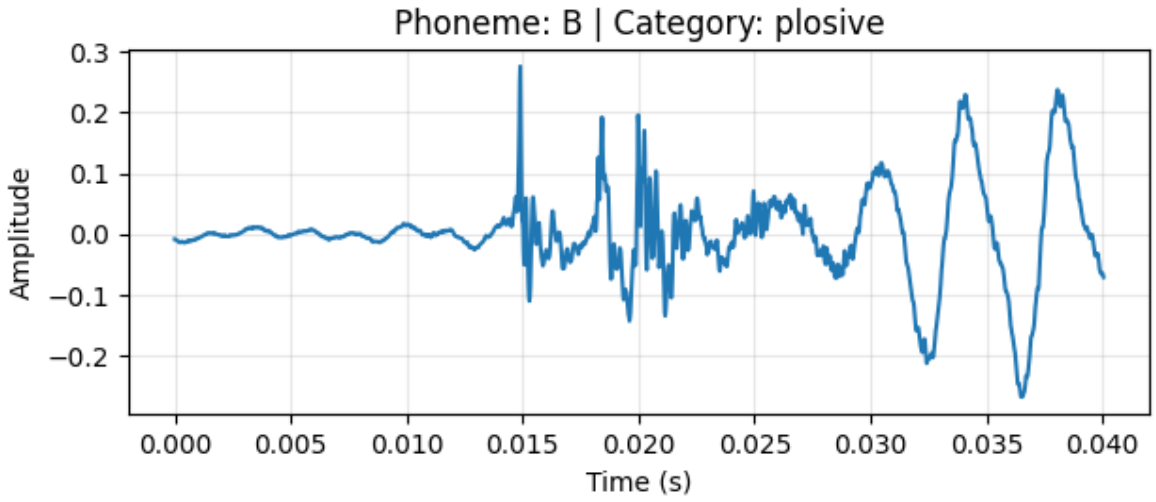

Phoneme: E | Category: vowel
Amplitude
Time (s)

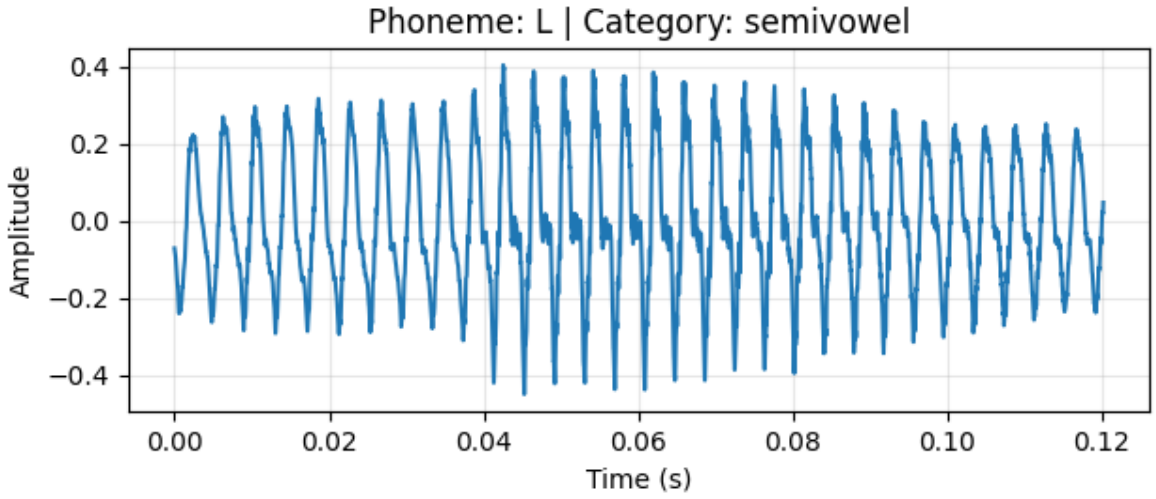

Phoneme: D | Category: plosive
Amplitude
Time (s)

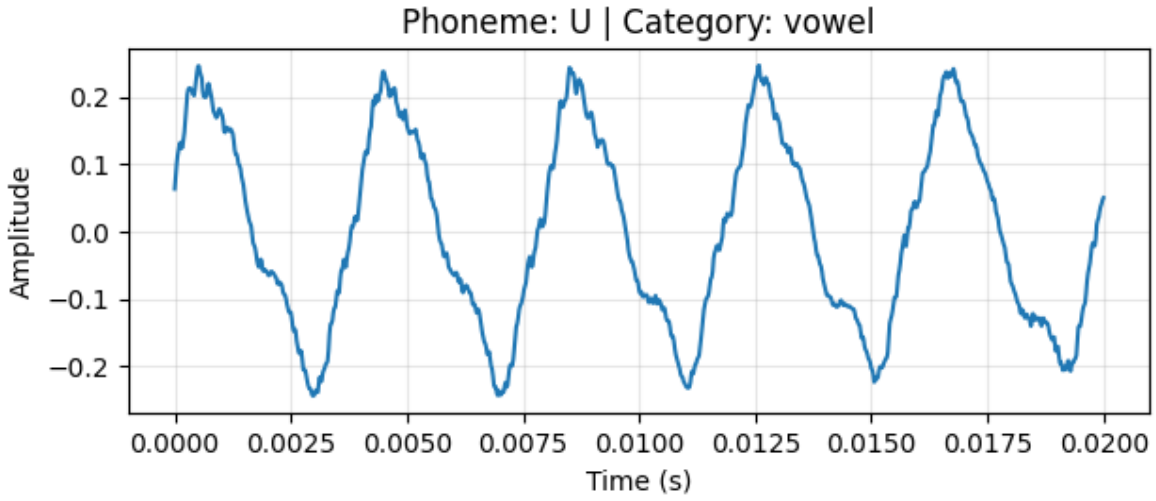

Phoneme: T | Category: plosive
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200

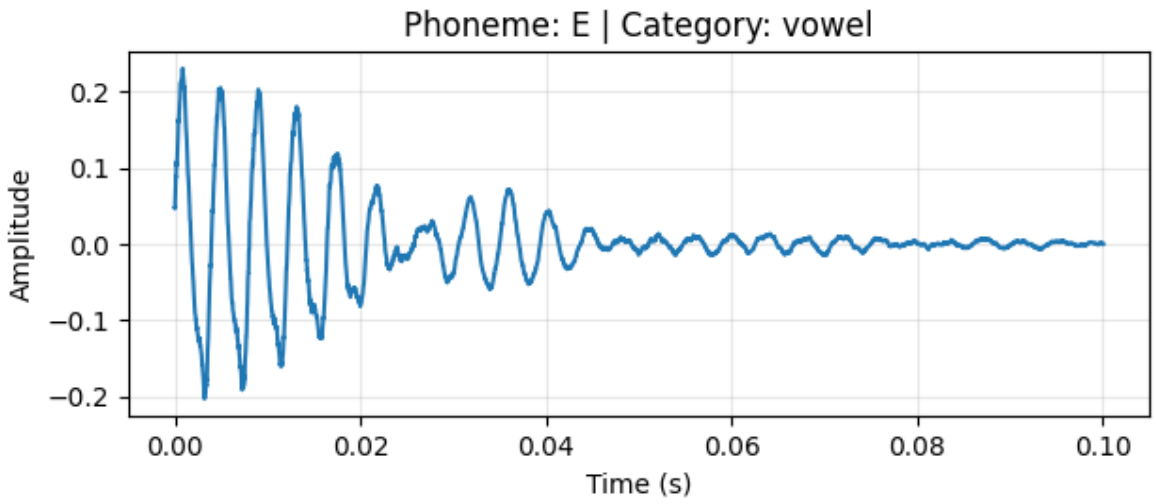

Phoneme: H | Category: fricative
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040

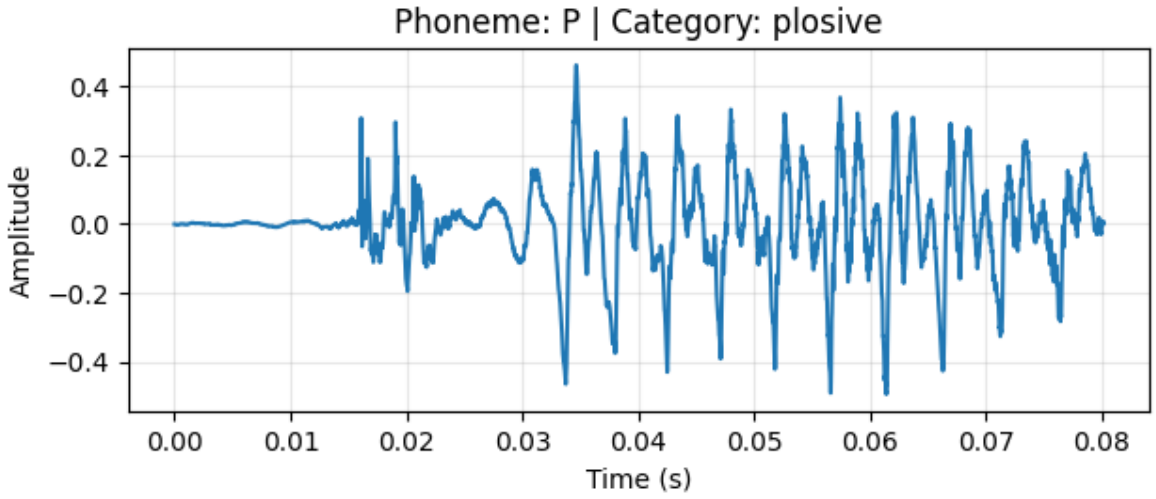

Phoneme: E | Category: vowel
Amplitude
Time (s)
0.1
0.0
−0.1
0.00
0.01
0.02
0.03
0.04
0.05
0.06

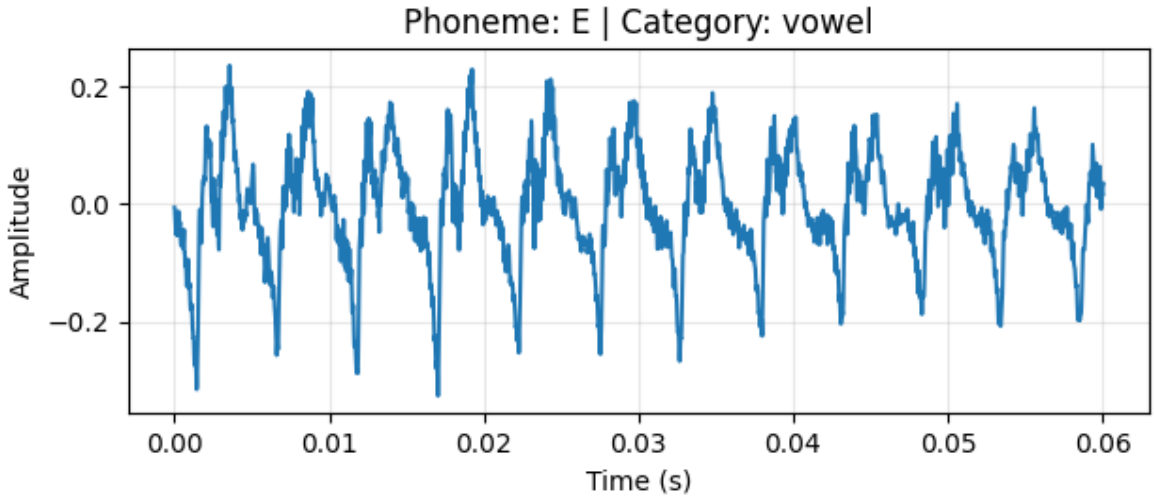

Phoneme: B | Category: plosive
Amplitude
Time (s)
0.3
0.2
0.1
0.0
−0.1
−0.2
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040

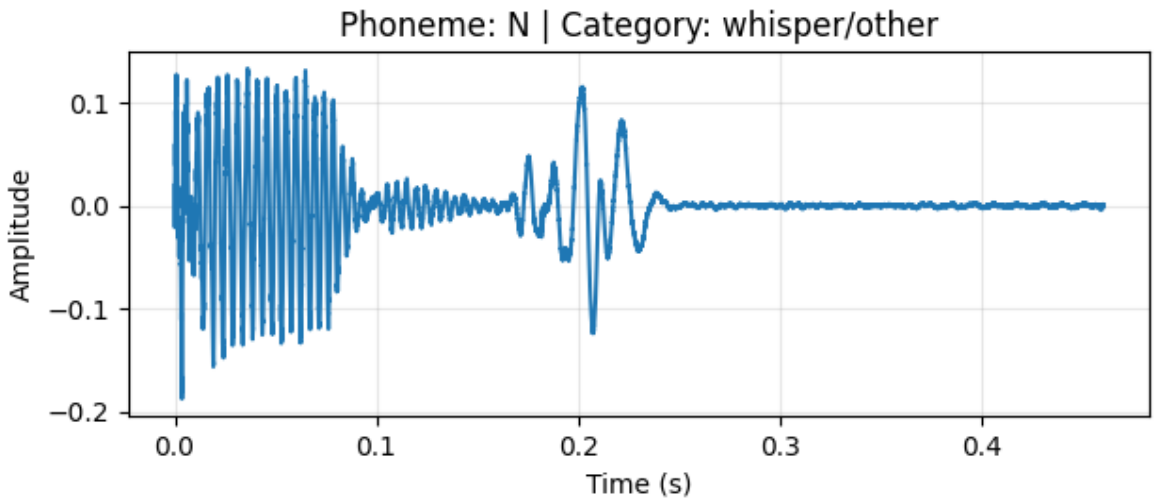

Phoneme: L | Category: semivowel
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.00
0.02
0.04
0.06
0.08
0.10
0.12

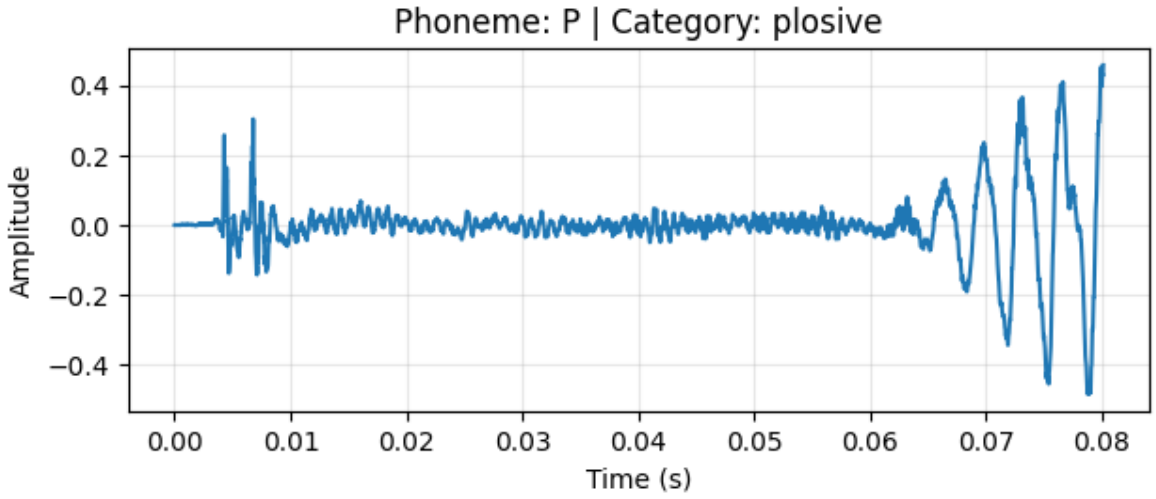

Phoneme: U | Category: vowel
Amplitude
Time (s)
0.2
0.1
0.0
−0.1
−0.2
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200

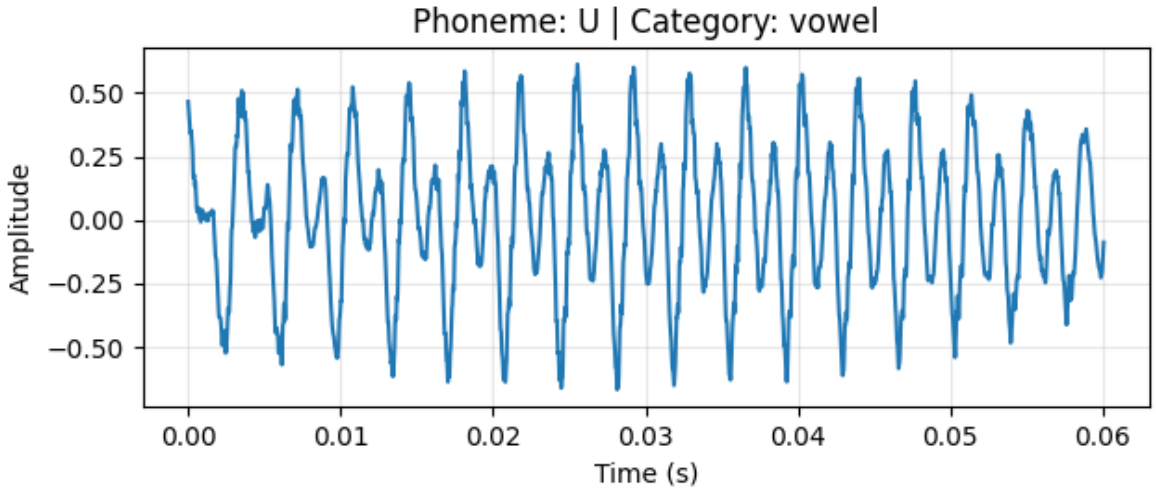

Phoneme: E | Category: vowel
Amplitude
0.2
0.1
0.0
−0.1
−0.2
0.00
0.02
0.04
0.06
0.08
0.10
Time (s)

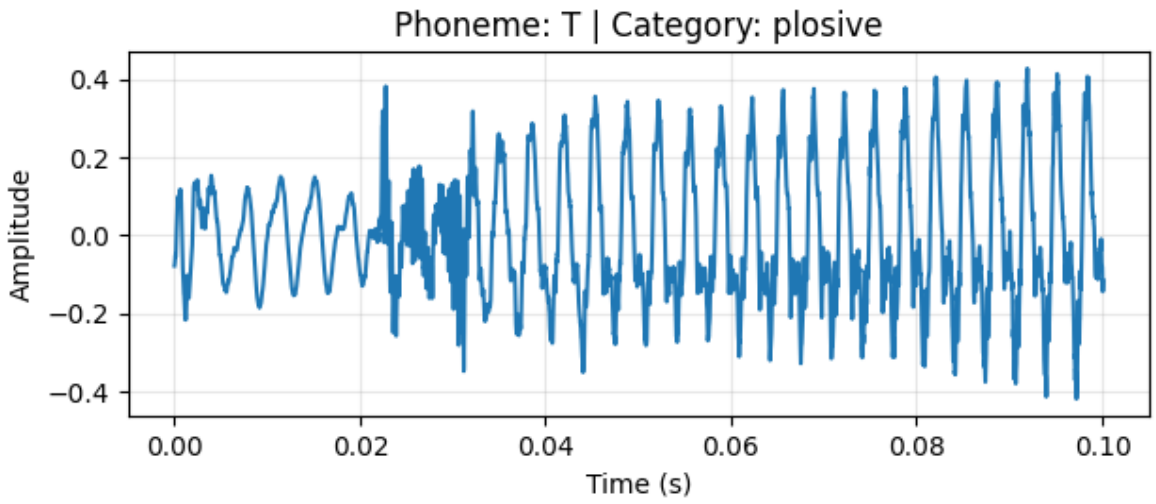

Phoneme: P | Category: plosive
Amplitude
0.4
0.2
0.0
−0.2
−0.4
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

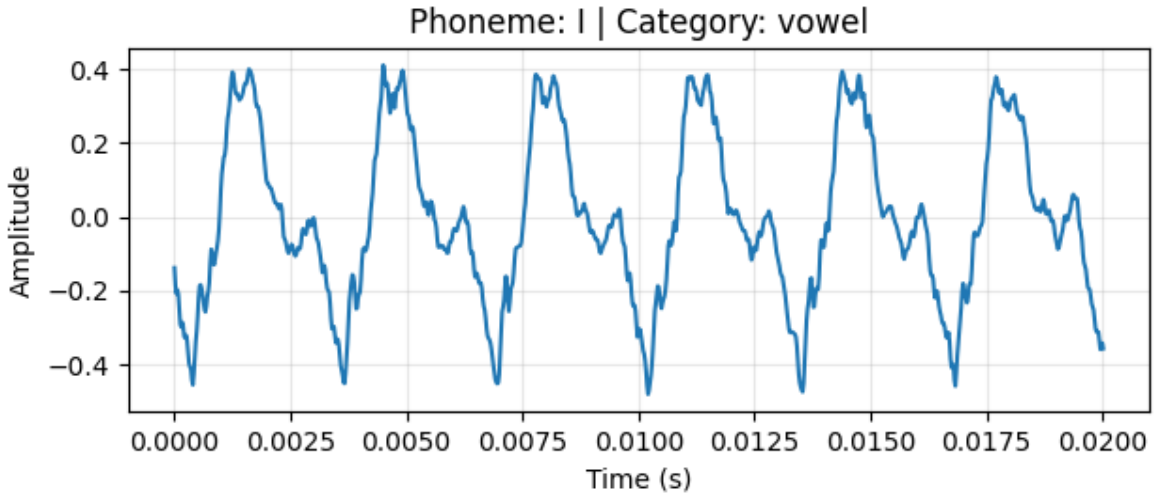

Phoneme: E | Category: vowel
Amplitude
0.2
0.0
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Time (s)

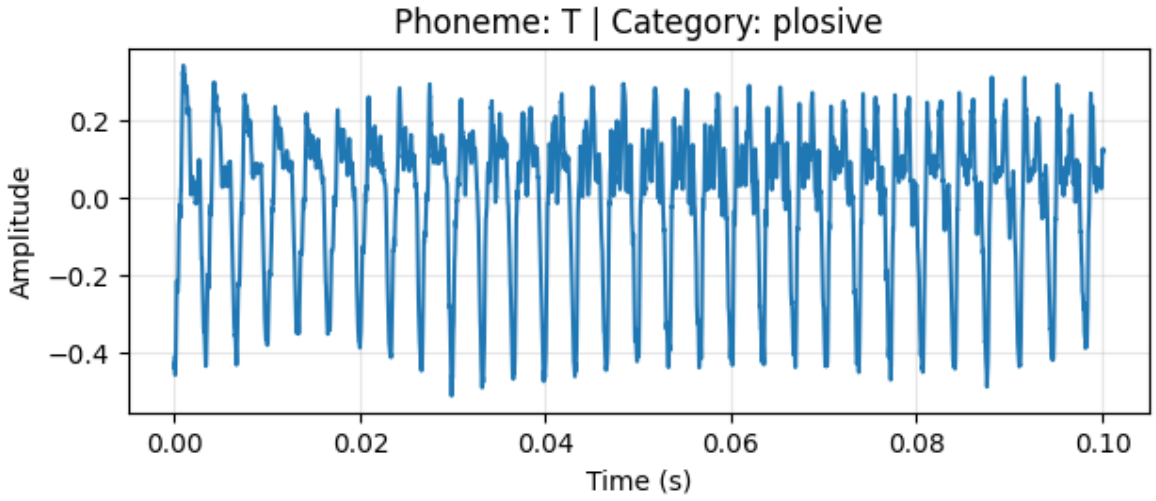

Phoneme: N | Category: whisper/other
Amplitude
0.1
0.0
−0.1
−0.2
0.0
0.1
0.2
0.3
0.4
Time (s)

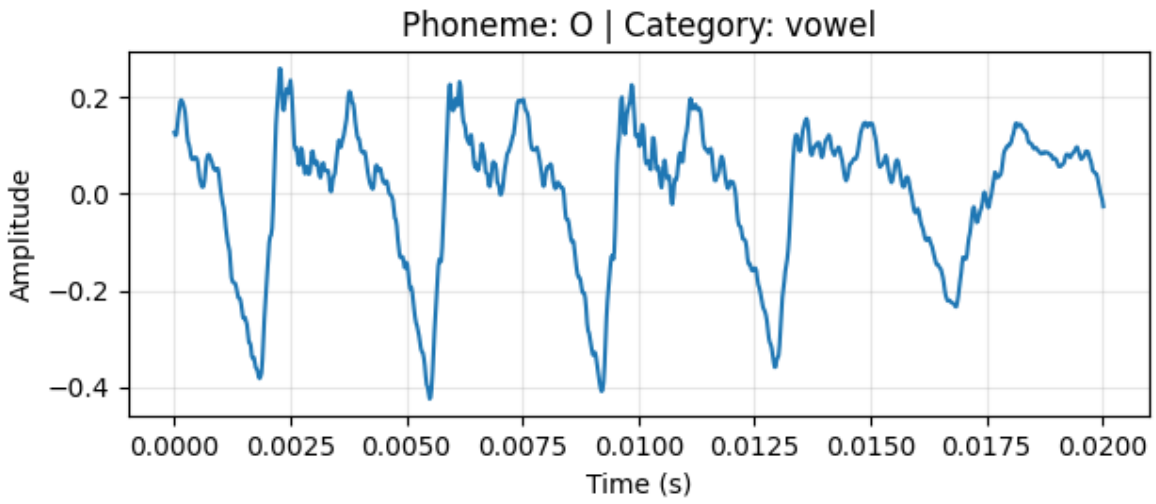

Phoneme: P | Category: plosive
Amplitude
0.4
0.2
0.0
−0.2
−0.4
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

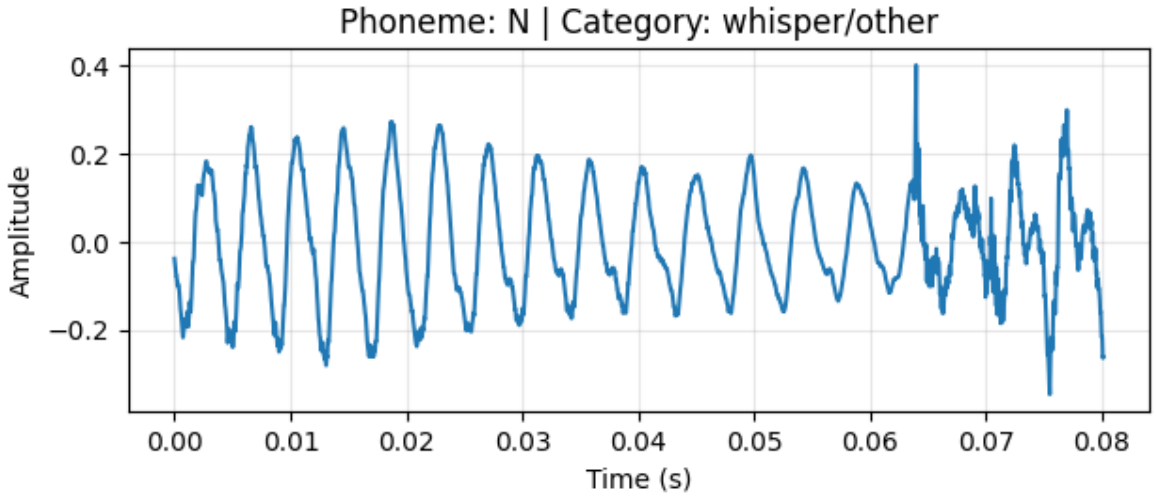

Phoneme: U | Category: vowel
Amplitude
0.50
0.25
0.00
−0.25
−0.50
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Time (s)

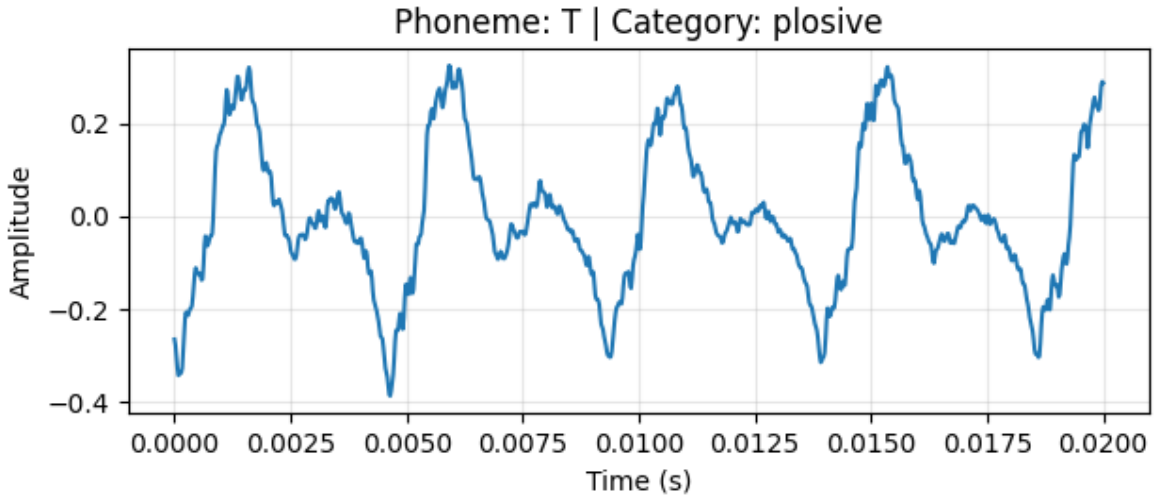

Phoneme: T | Category: plosive
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.00
0.02
0.04
0.06
0.08
0.10

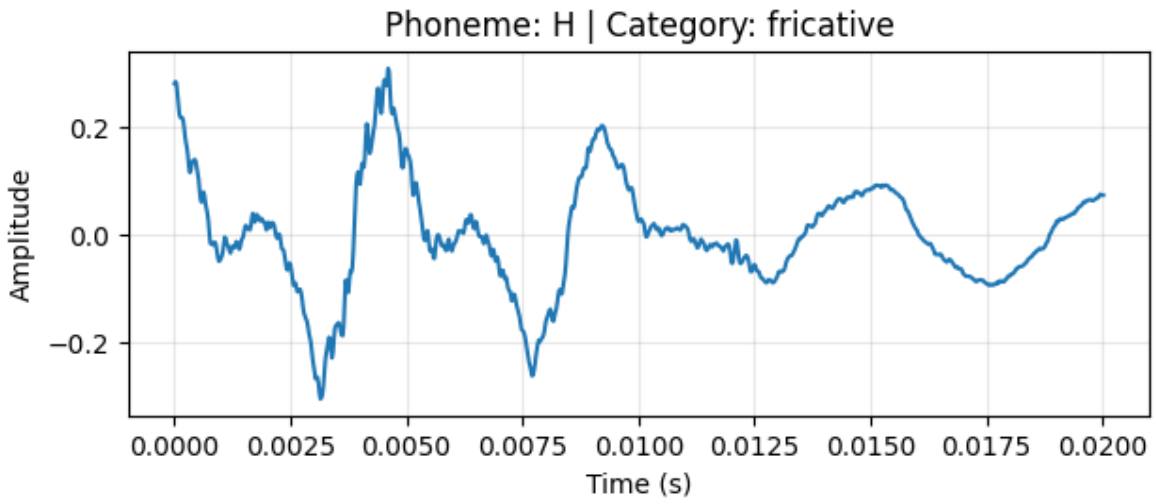

Phoneme: I | Category: vowel
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200

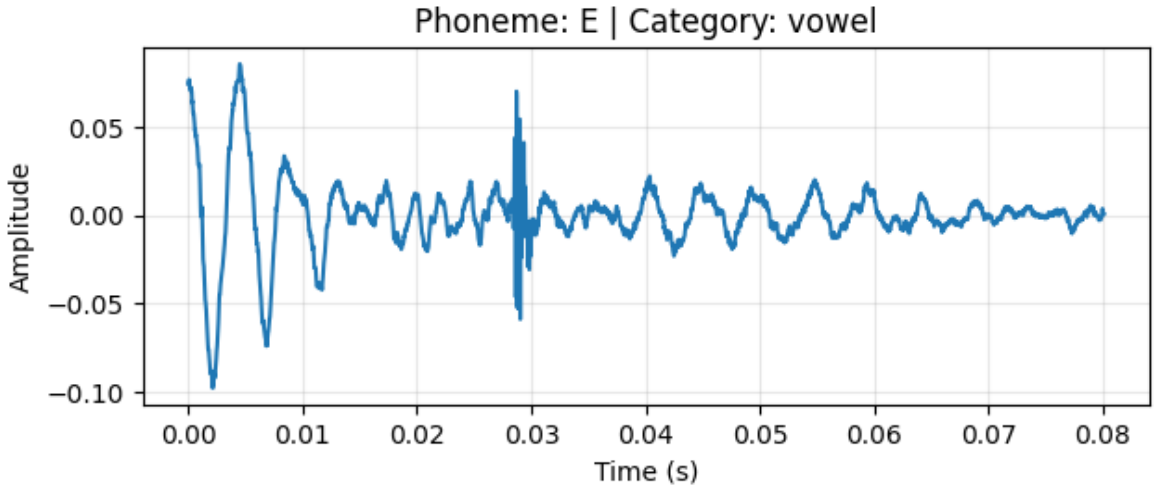

Phoneme: T | Category: plosive
Amplitude
Time (s)
0.2
0.0
−0.2
−0.4
0.00
0.02
0.04
0.06
0.08
0.10

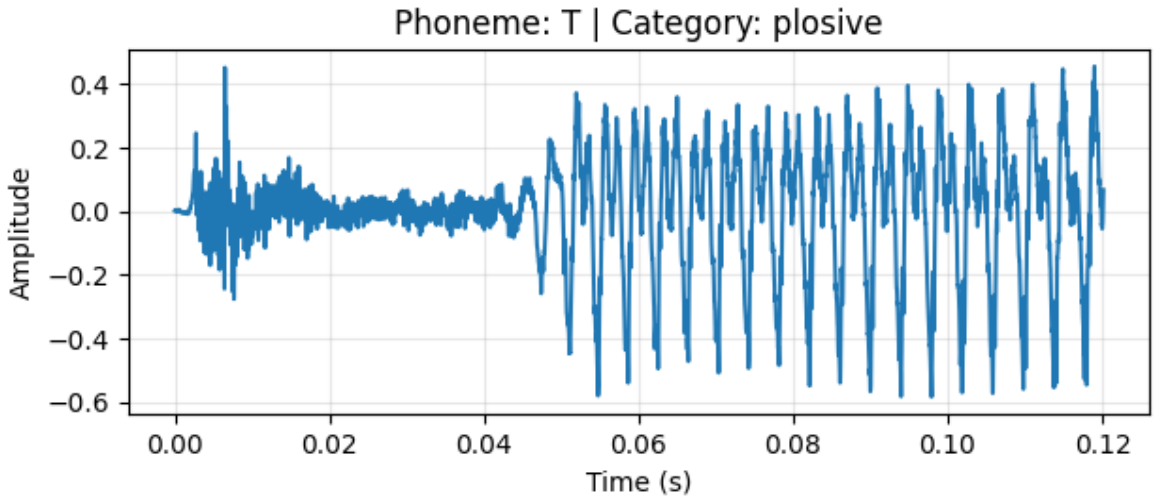

Phoneme: O | Category: vowel
Amplitude
0.2
0.0
−0.2
−0.4
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
Time (s)

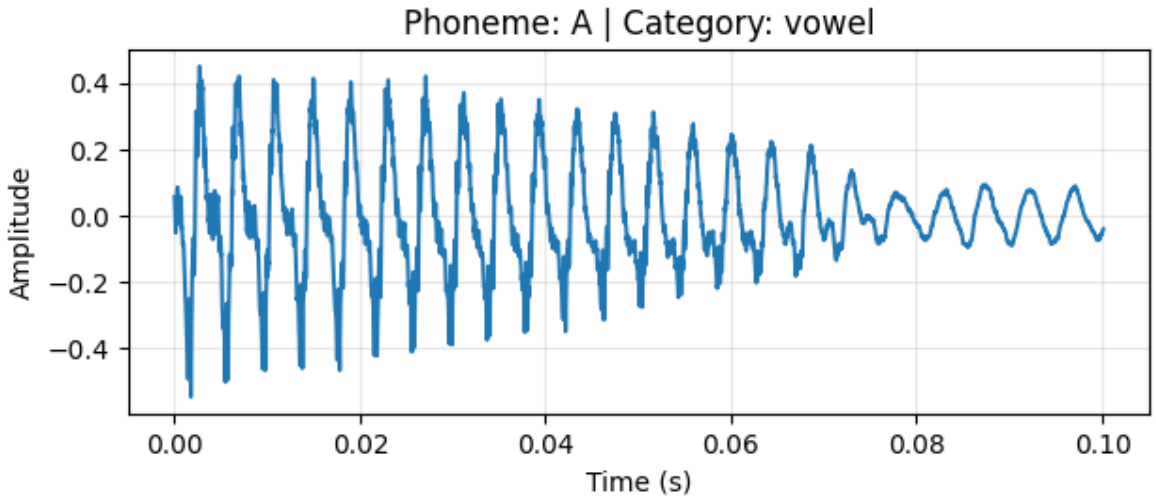

Phoneme: N | Category: whisper/other
Amplitude
0.4
0.2
0.0
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

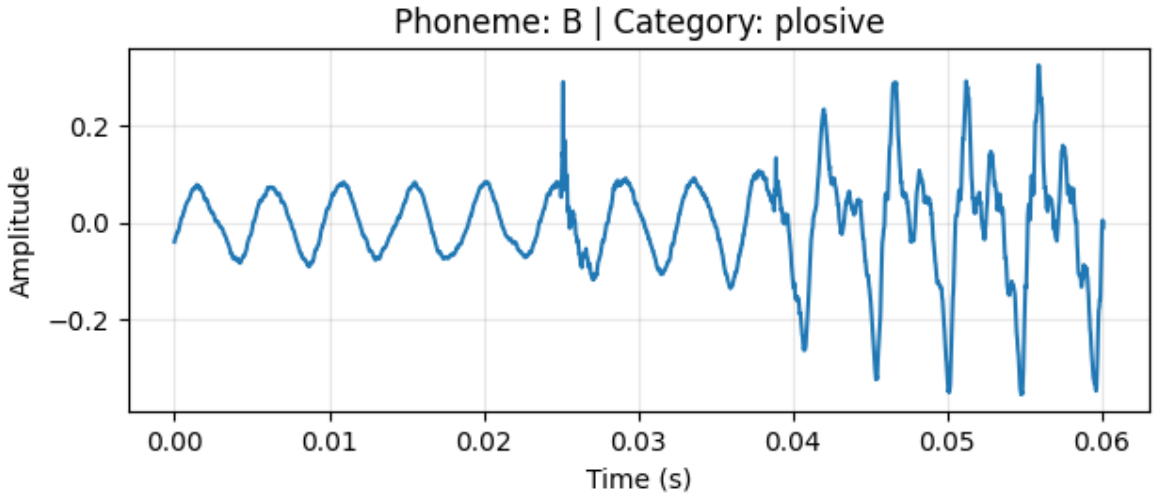

Phoneme: T | Category: plosive
Amplitude
0.2
0.0
−0.2
−0.4
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
Time (s)

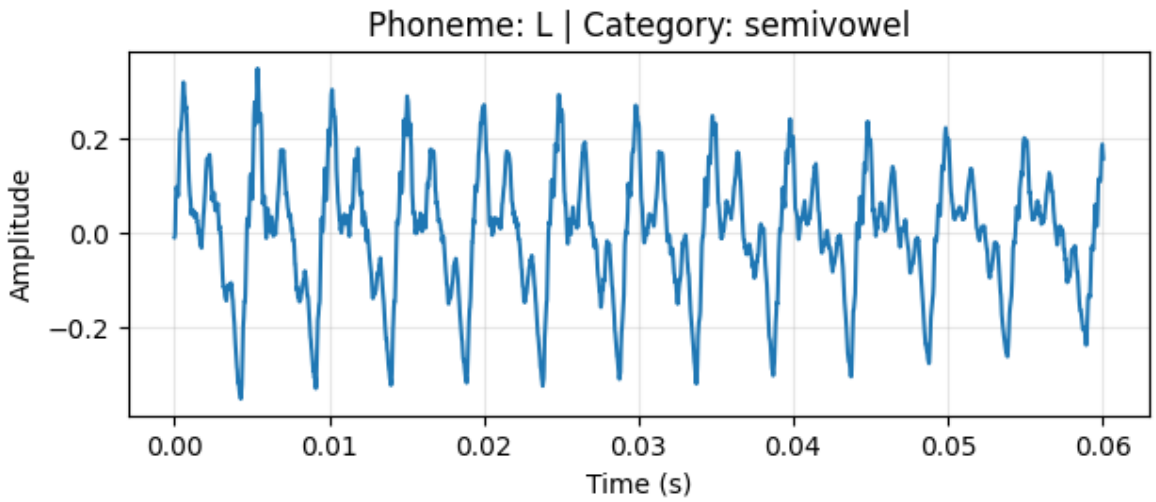

Phoneme: H | Category: fricative
Amplitude
0.2
0.0
−0.2
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
Time (s)

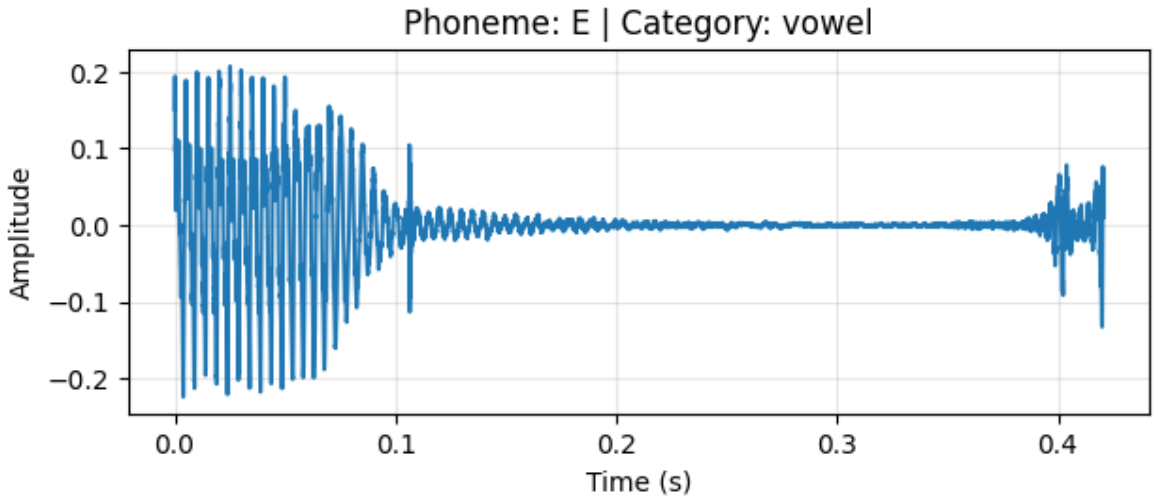

Phoneme: E | Category: vowel
Amplitude
0.05
0.00
−0.05
−0.10
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

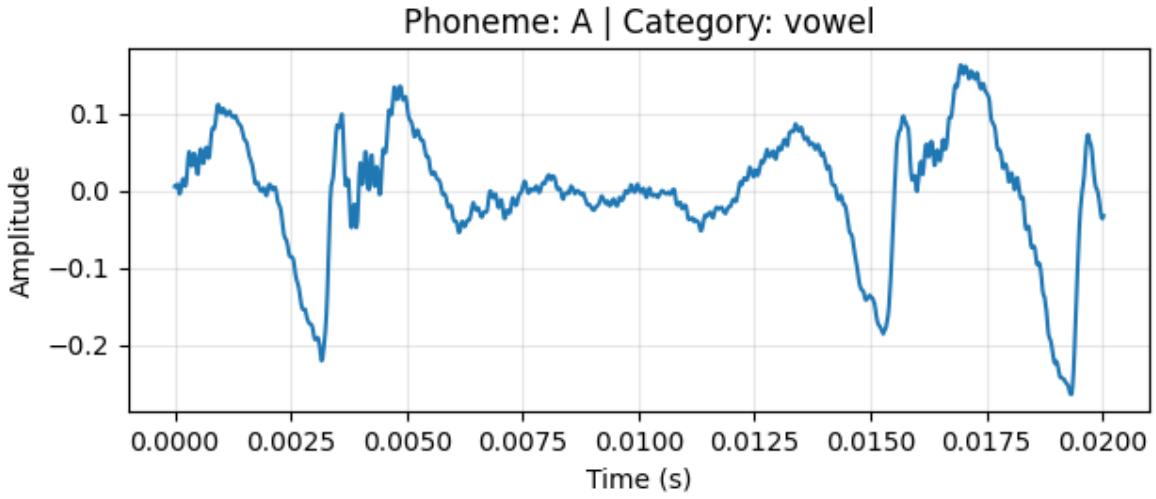

Phoneme: T | Category: plosive
Amplitude
0.4
0.2
0.0
−0.2
−0.4
−0.6
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Time (s)

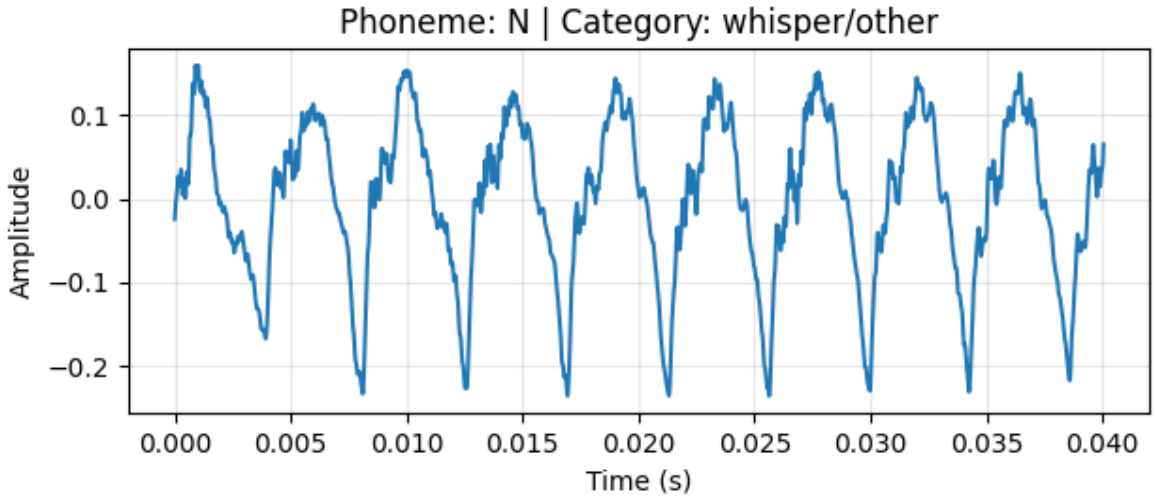

Phoneme: A | Category: vowel
Amplitude
0.4
0.2
0.0
−0.2
−0.4
0.00
0.02
0.04
0.06
0.08
0.10
Time (s)

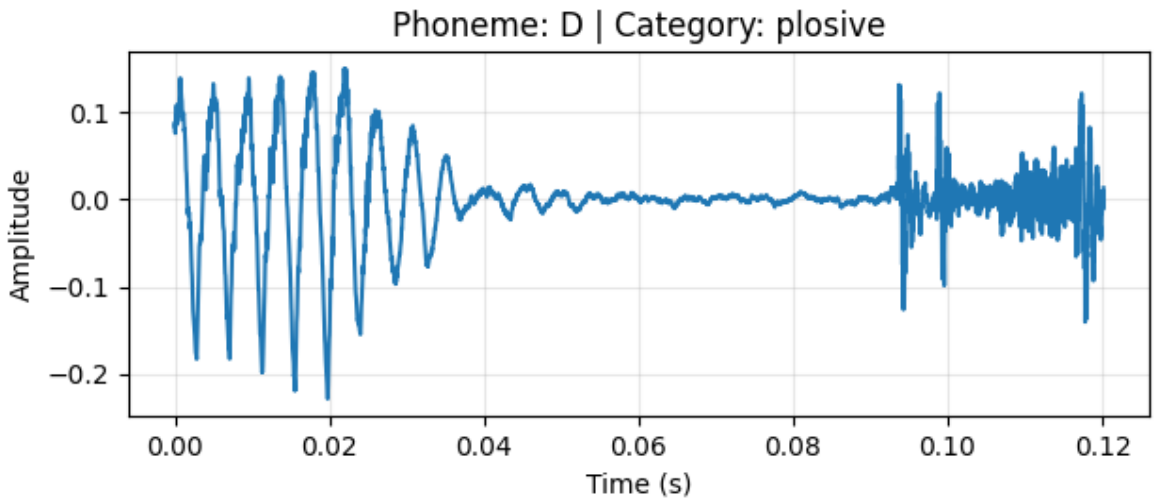

Phoneme: B | Category: plosive
Amplitude
0.2
0.0
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Time (s)

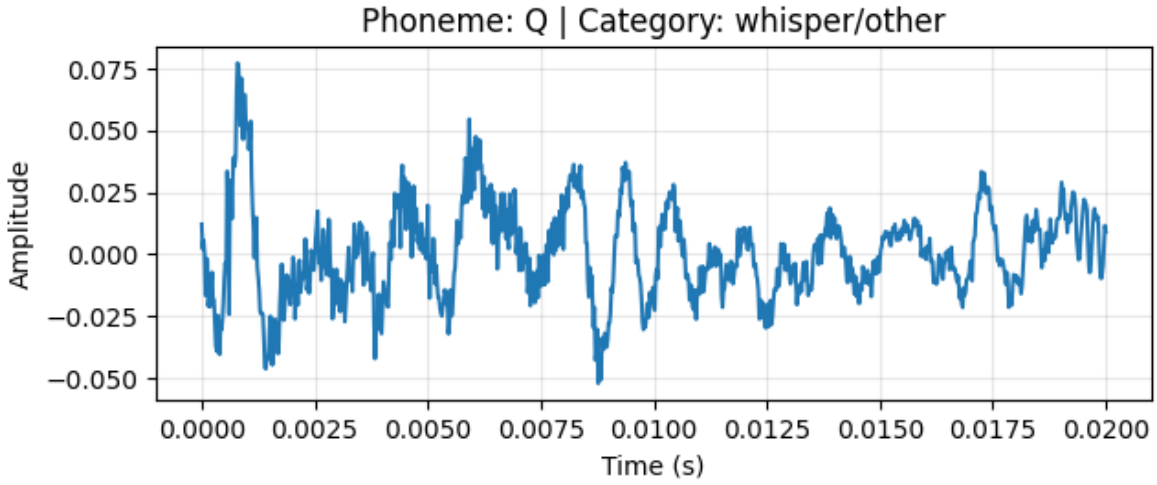

Phoneme: L | Category: semivowel
Amplitude
0.2
0.0
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Time (s)

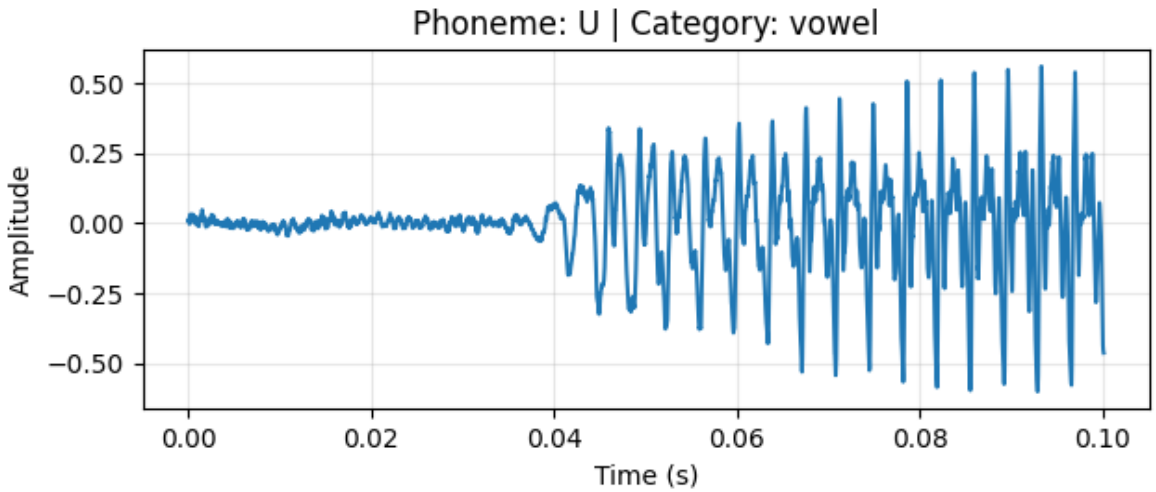

Phoneme: E | Category: vowel
Amplitude
Time (s)

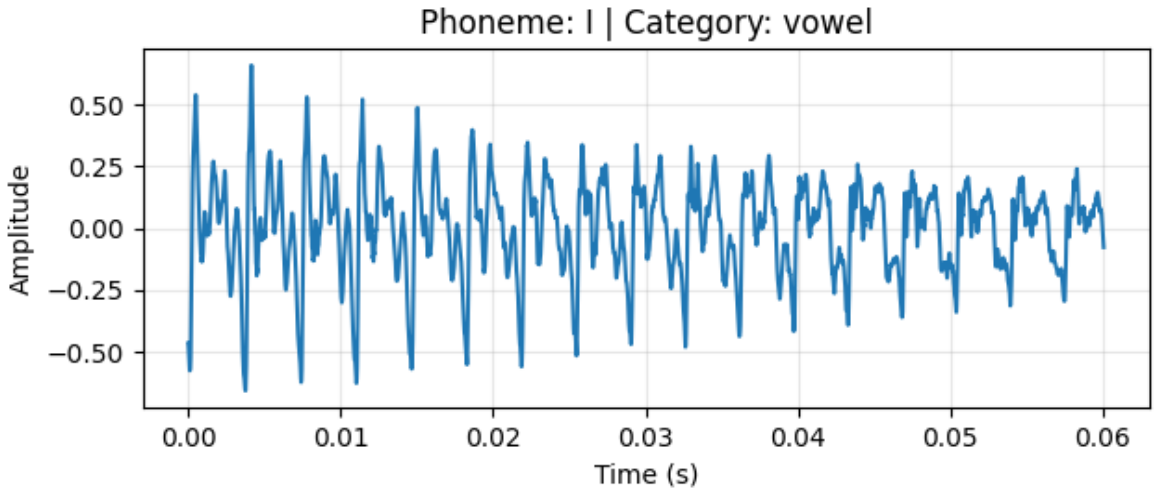

Phoneme: A | Category: vowel
Amplitude
Time (s)

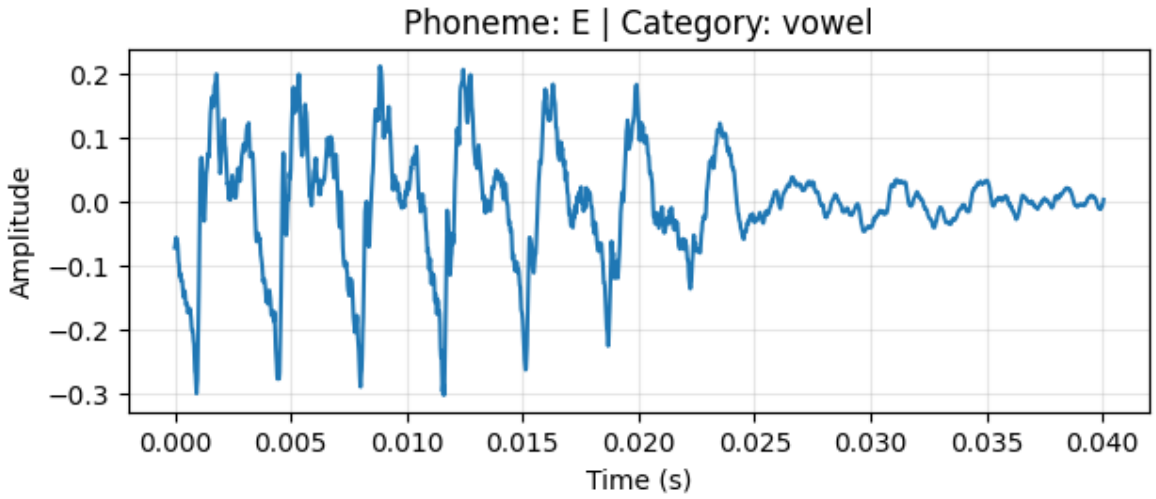

Phoneme: N | Category: whisper/other
Amplitude
Time (s)

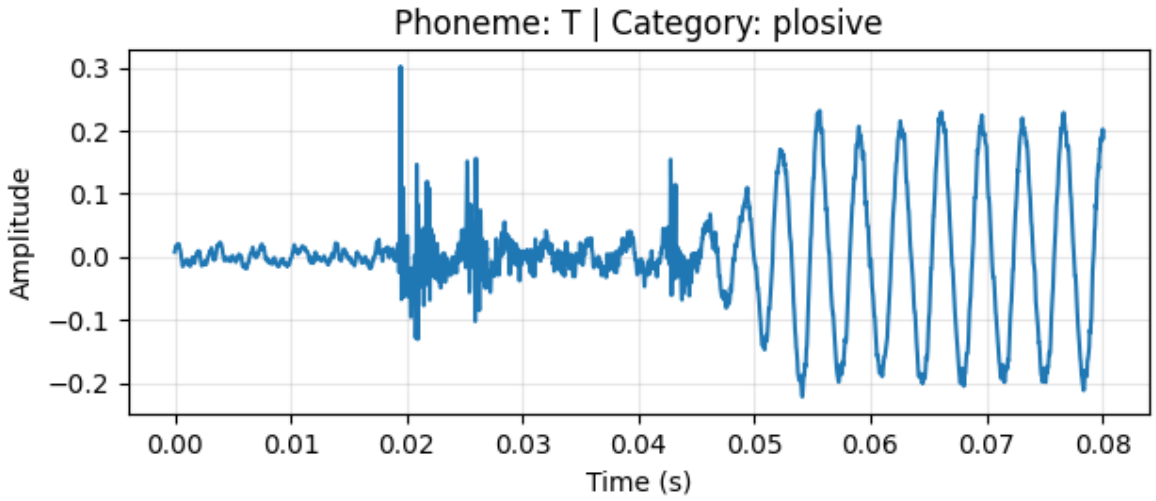

Phoneme: D | Category: plosive
Amplitude
0.1
0.0
−0.1
−0.2
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Time (s)

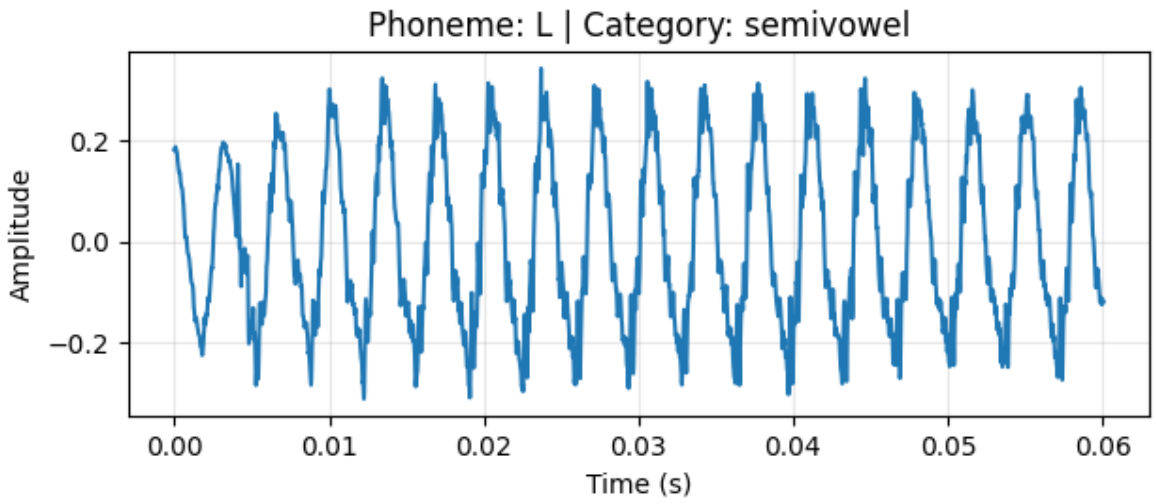

Phoneme: Q | Category: whisper/other
Amplitude
0.075
0.050
0.025
0.000
−0.025
−0.050
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
Time (s)

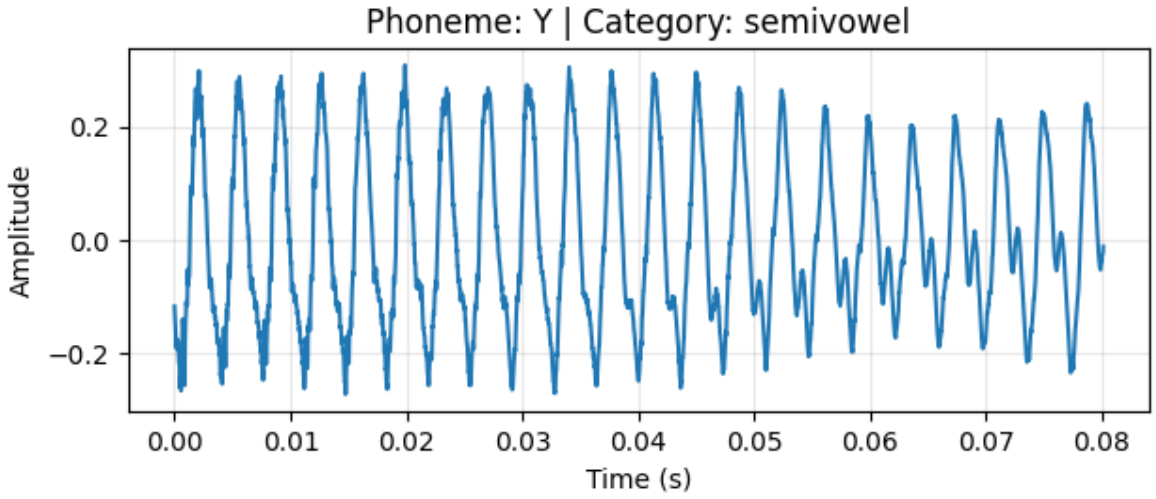

Phoneme: U | Category: vowel
Amplitude
0.50
0.25
0.00
−0.25
−0.50
0.00
0.02
0.04
0.06
0.08
0.10
Time (s)

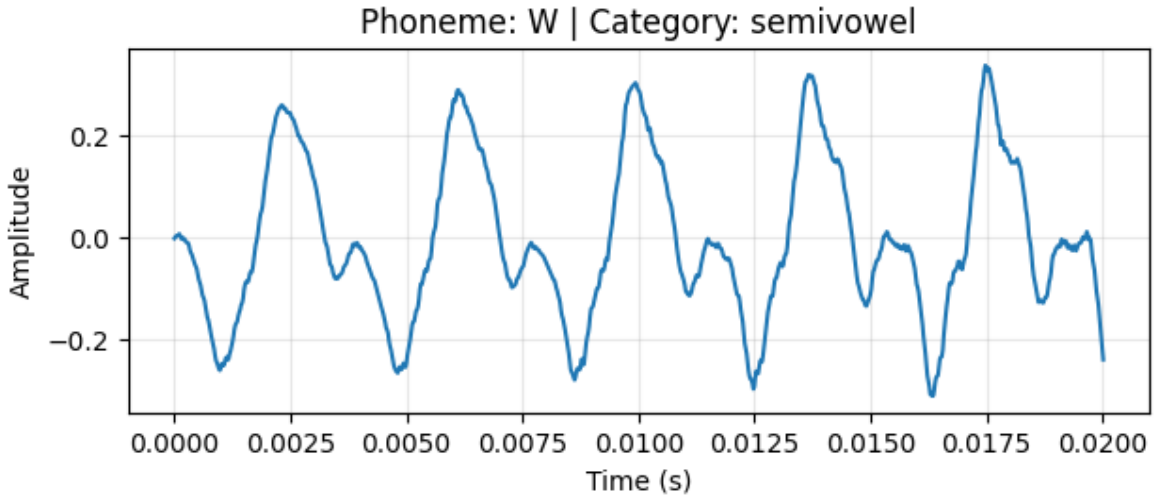

Phoneme: I | Category: vowel
Amplitude
0.50
0.25
0.00
−0.25
−0.50
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Time (s)

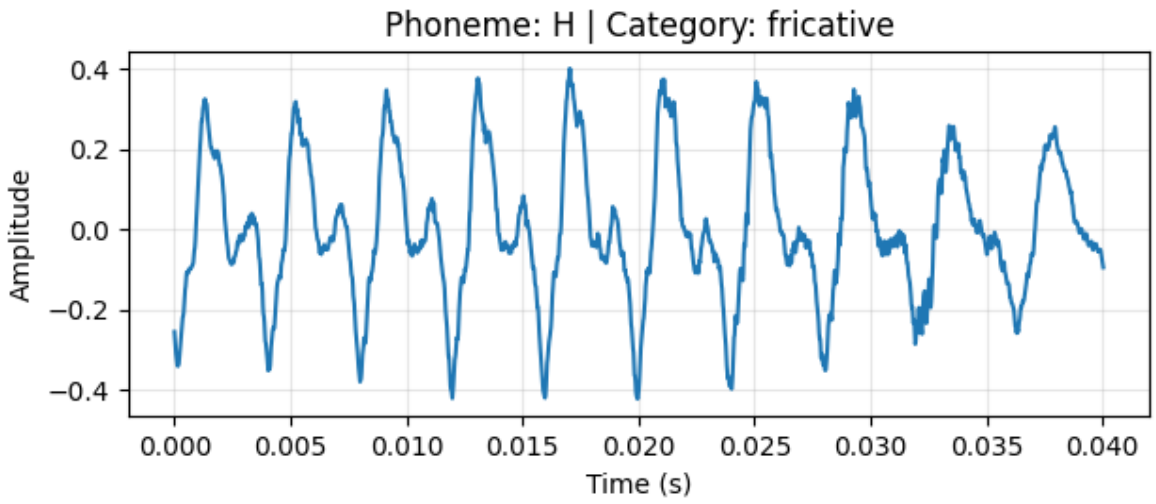

Phoneme: E | Category: vowel
Amplitude
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
Time (s)

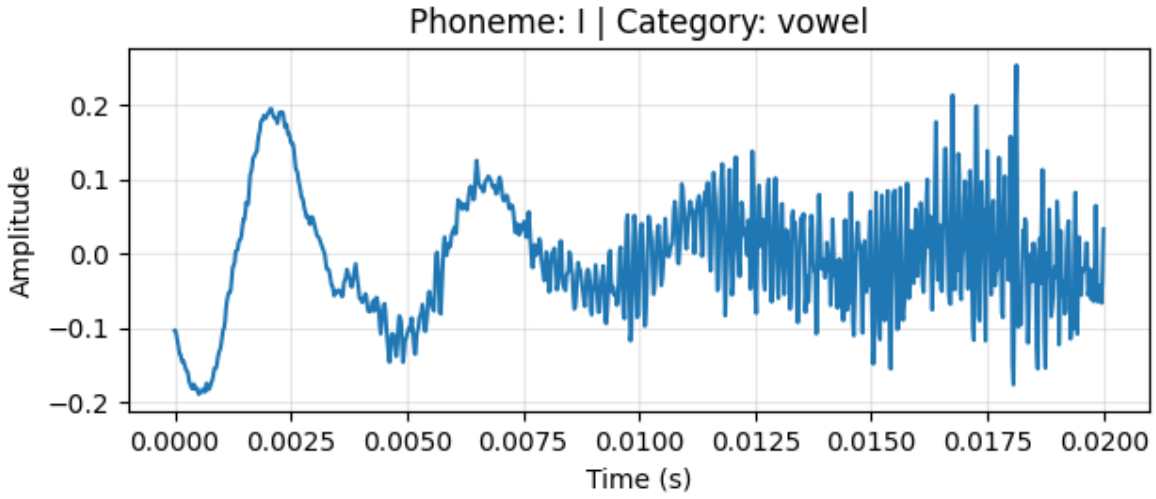

Phoneme: T | Category: plosive
Amplitude
0.3
0.2
0.1
0.0
−0.1
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

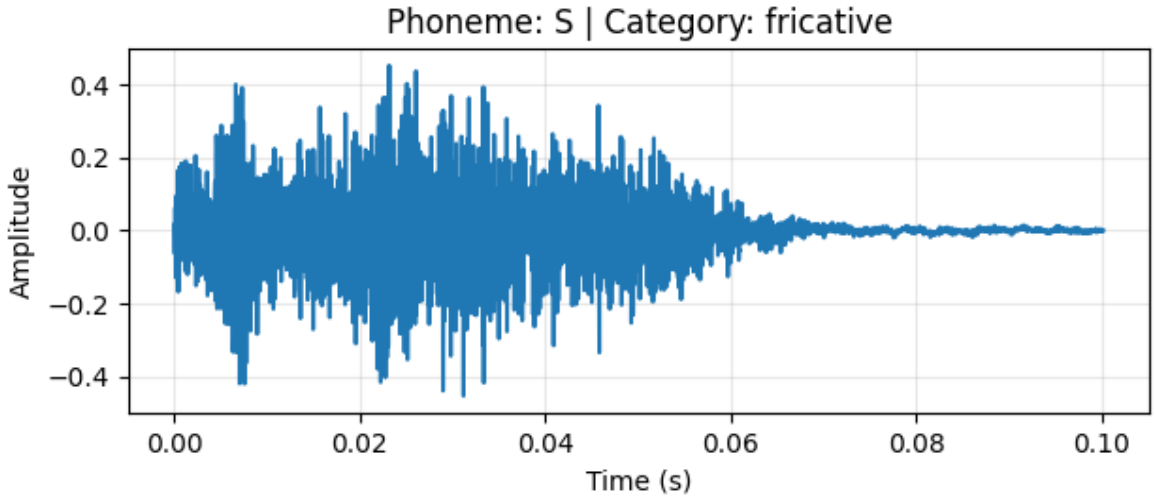

Phoneme: L | Category: semivowel
Amplitude
0.2
0.0
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
Time (s)

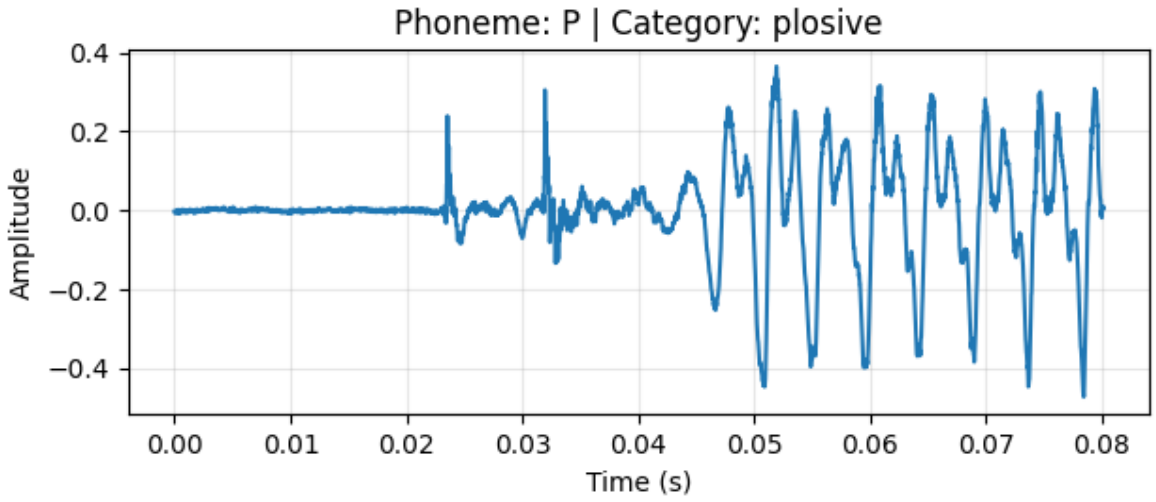

Phoneme: Y | Category: semivowel
Amplitude
0.2
0.0
−0.2
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

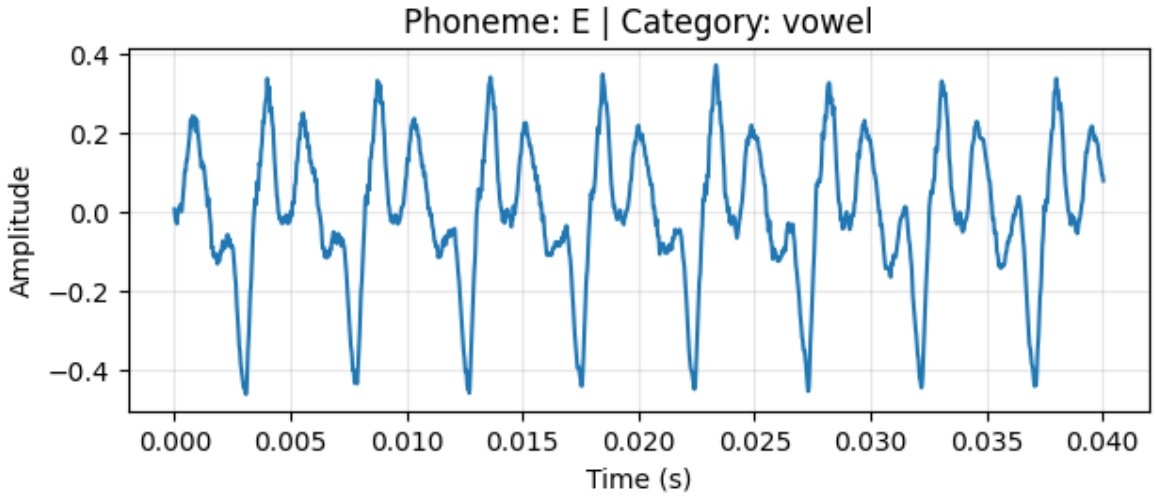

Phoneme: W | Category: semivowel
Amplitude
0.2
0.0
−0.2
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
Time (s)

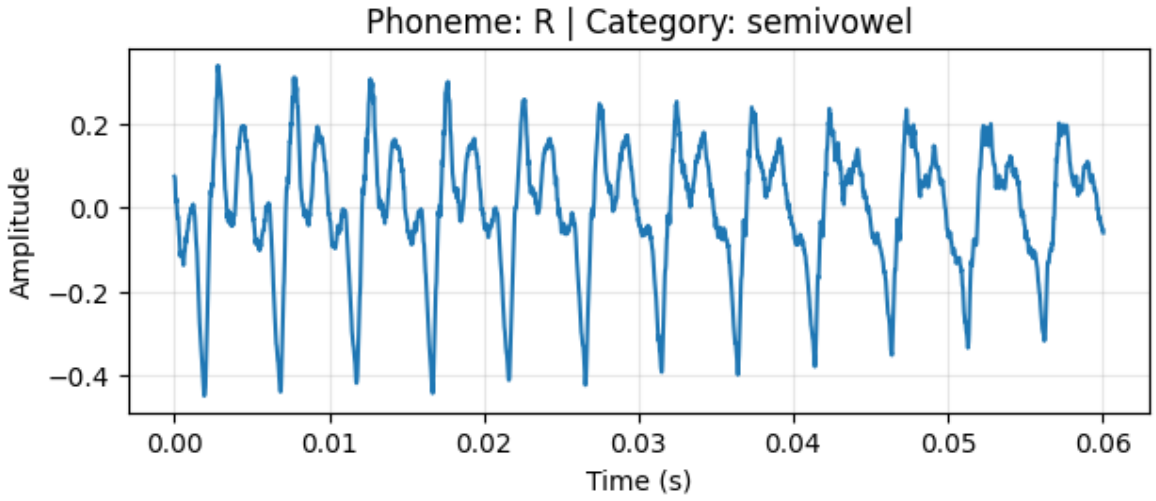

Phoneme: H | Category: fricative
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040

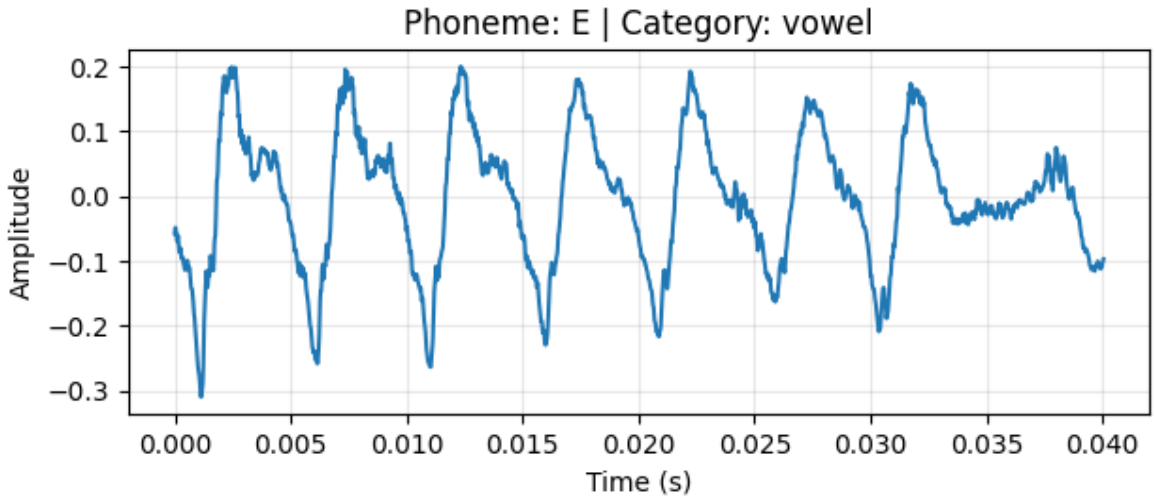

Phoneme: I | Category: vowel
Amplitude
Time (s)
0.2
0.1
0.0
−0.1
−0.2
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200

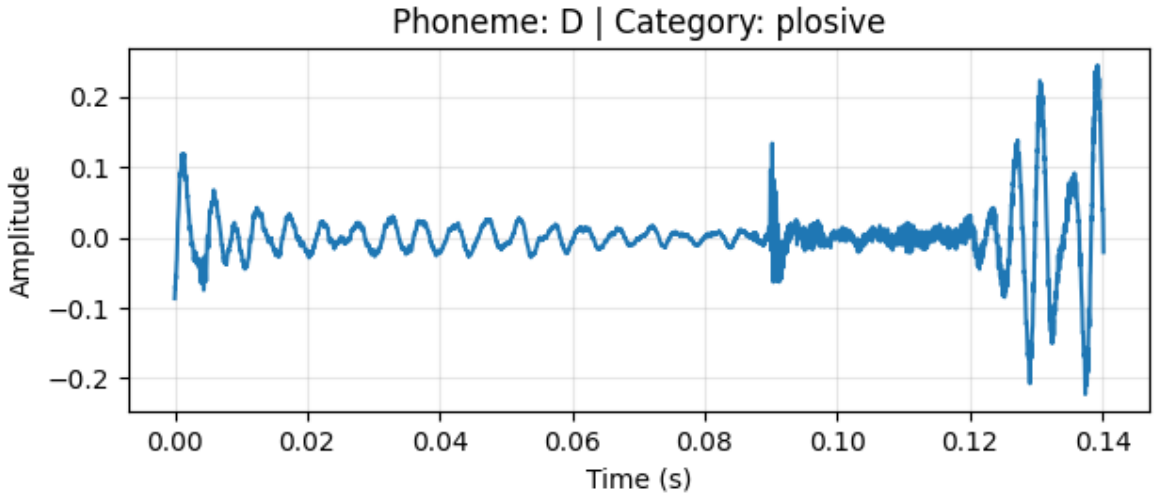

Phoneme: S | Category: fricative
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.00
0.02
0.04
0.06
0.08
0.10

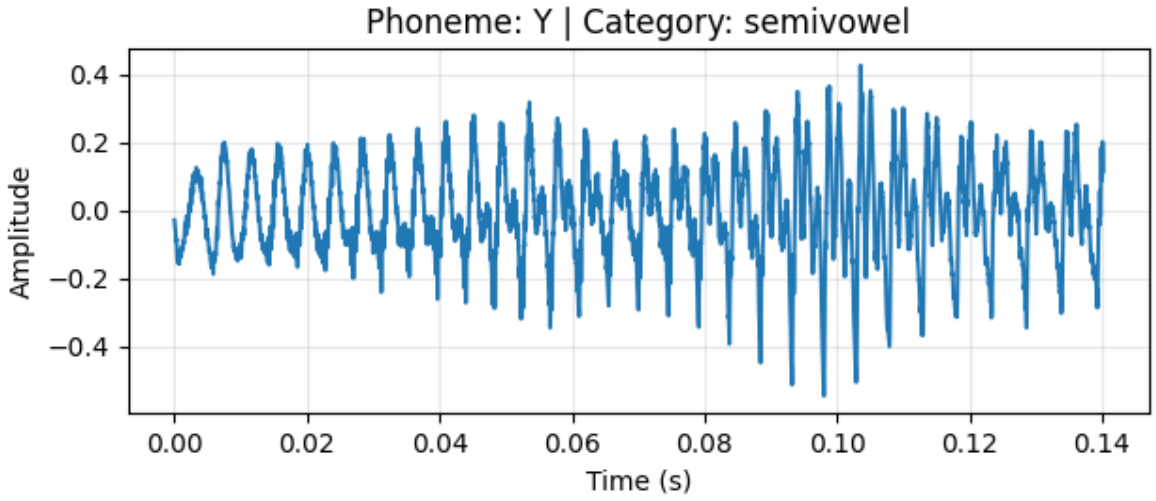

Phoneme: P | Category: plosive
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08

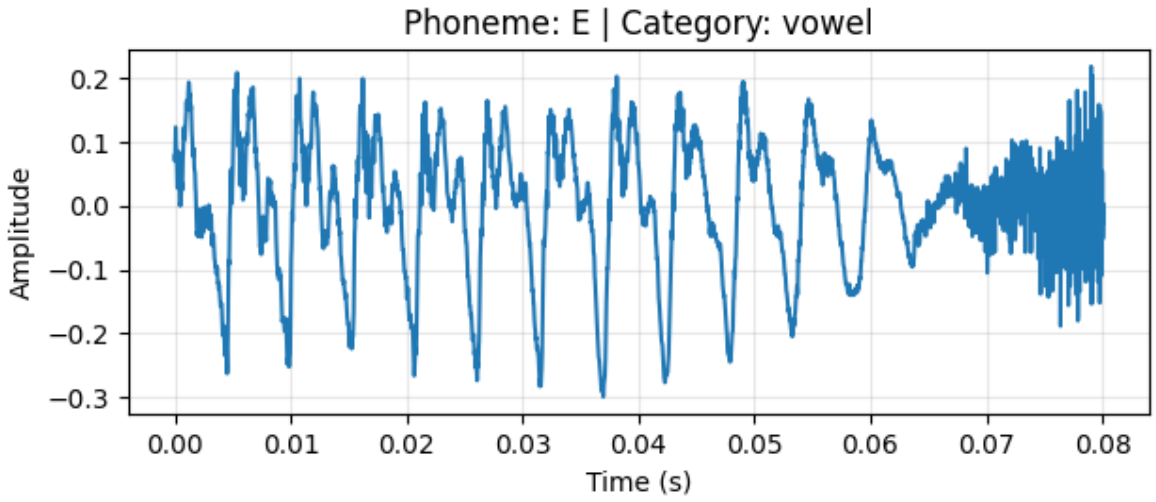

Phoneme: E | Category: vowel
Amplitude
Time (s)
0.4
0.2
0.0
−0.2
−0.4
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040

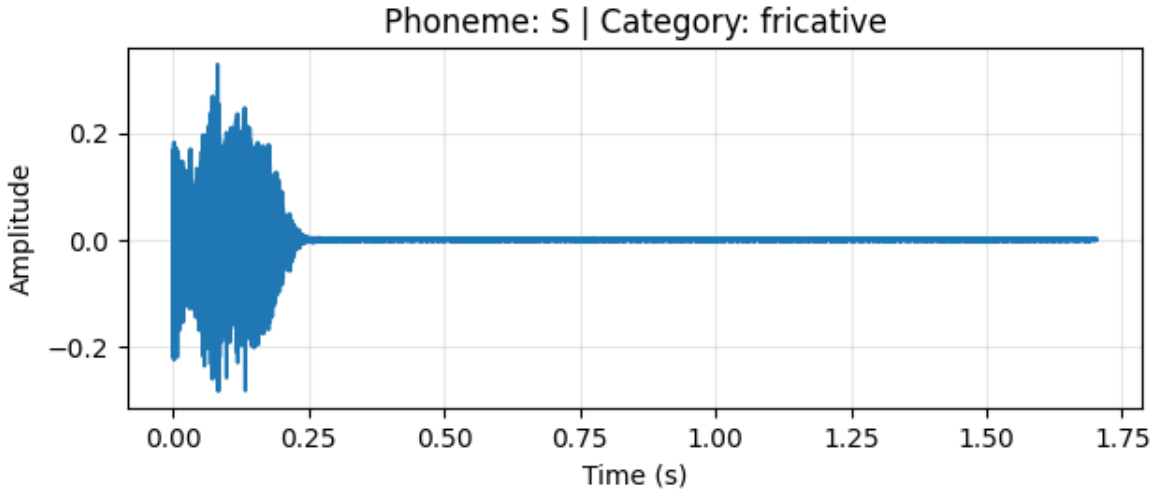

Phoneme: R | Category: semivowel
Amplitude
Time (s)
0.2
0.0
−0.2
−0.4
0.00
0.01
0.02
0.03
0.04
0.05
0.06


Phoneme: E | Category: vowel
Amplitude
Time (s)


Phoneme: D | Category: plosive
Amplitude
Time (s)


Phoneme: Y | Category: semivowel
Amplitude
Time (s)



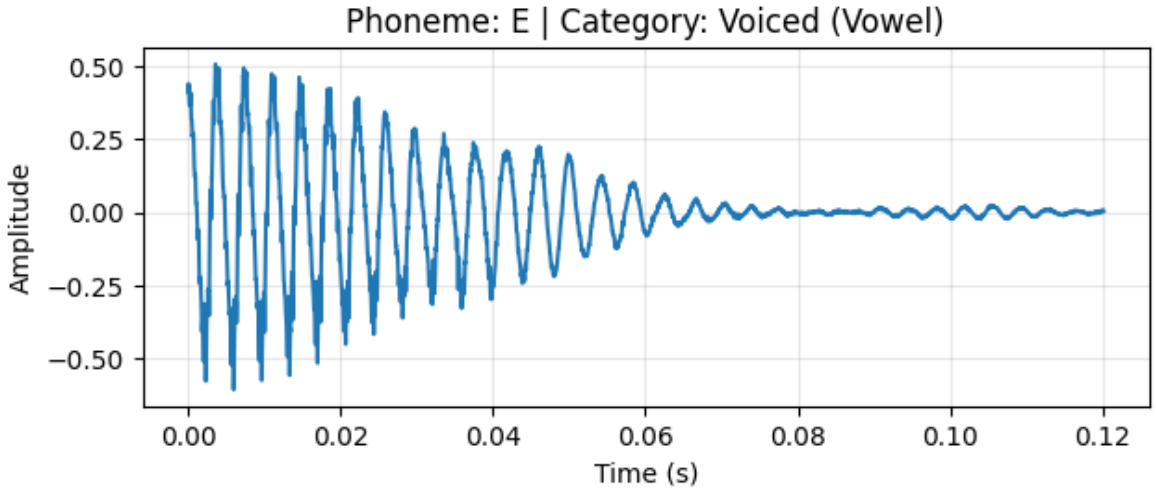




## 9. Voiced vs Unvoiced Comparison

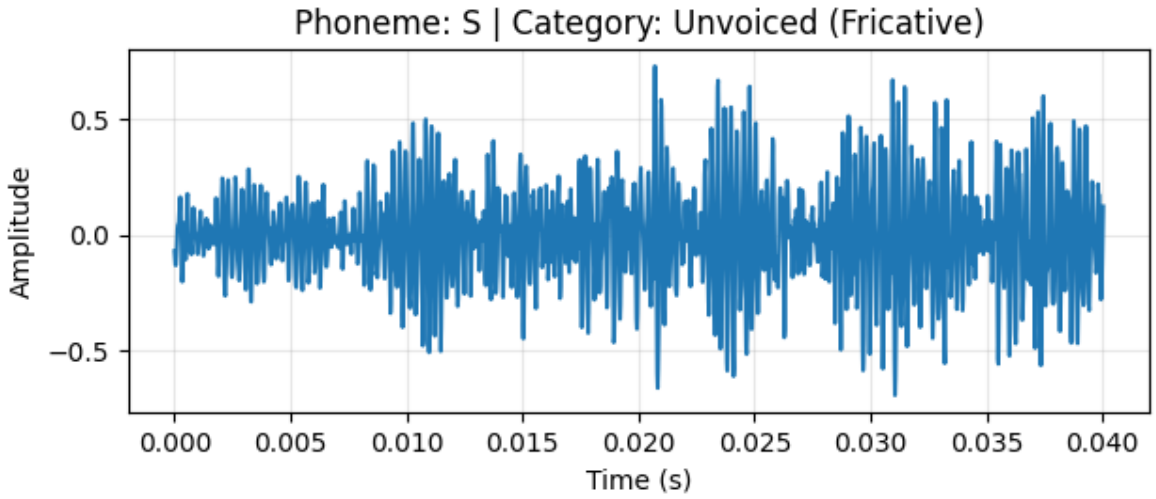
In [9]:
```
def get_category(p):
    vowels = ['A', 'E', 'I', 'O', 'U']
    fricatives = ['S', 'F', 'Z', 'H']
    char = p["phoneme"].upper()
    if char in vowels: return "vowel"
    if char in fricatives: return "fricative"
    return "other"

voiced = next((p for p in phoneme_intervals if get_category(p) == "vowel"), None
unvoiced = next((p for p in phoneme_intervals if get_category(p) == "fricative")

if voiced and unvoiced:
    print(f"Voiced segment: Phoneme '{voiced['phoneme']}' (Category: {voiced['ca
    plot_phoneme(
        waveform[0, voiced["start_sample"]:voiced["end_sample"]].numpy(),
        sample_rate,
        voiced["phoneme"],
        "Voiced (Vowel)"
    )

    print(f"Unvoiced segment: Phoneme '{unvoiced['phoneme']}' (Category: {unvoic
    plot_phoneme(
        waveform[0, unvoiced["start_sample"]:unvoiced["end_sample"]].numpy(),
        sample_rate,
```

```
        unvoiced["phoneme"],
        "Unvoiced (Fricative)"
    )
else:
    print("Error: Could not find both a voiced and unvoiced segment.")
    print("Detected letters:", [p['phoneme'] for p in phoneme_intervals])
```

```
Voiced segment: Phoneme 'E' (Category: vowel)
```




```
Unvoiced segment: Phoneme 'S' (Category: fricative)
```




# Comparison Analysis

**Observed differences between voiced and unvoiced phonemes:**

**Periodicity:**

- Voiced phonemes (vowels) show clear periodic patterns in the waveform due to regular vocal fold vibration
- Unvoiced phonemes (fricatives) display aperiodic, random fluctuations with no repeating pattern

**Amplitude patterns:**

- Voiced phonemes have more consistent amplitude with smooth envelope
- Unvoiced phonemes show irregular amplitude variations typical of turbulent airflow

**Presence or absence of noise-like structure:**

- Voiced phonemes: Absence of noise-like structure; waveform shows harmonic content
- Unvoiced phonemes: Strong noise-like structure; waveform resembles random noise from turbulent air passing through constriction

# Objective 2

To analyze fricative and approximant speech sounds and infer airflow turbulence characteristics.
Record the sentence: “She sees you.”
Perform the following:

- Record the given sentence at a sampling rate of 16 kHz, mono WAV format. Load the recorded speech signal.
- Identify regions corresponding to fricatives and approximants.
- Use the segmentation approach implemented in the previous lab to automatically extract phoneme-level segments.
- Isolate segments corresponding to fricatives (/s/, /ʃ/, /z/) and approximants (/j/).
- Plot the waveform of each extracted fricative and approximant segment.
- Observe qualitative differences in waveform smoothness, amplitude continuity, presence of noise-like fluctuations.
- Based on waveform observations, infer conceptually whether the airflow is turbulent (associated with higher effective Reynolds number) or Smooth/laminar (associated with lower effective Reynolds number).
- Compare fricatives and approximants in terms of waveform irregularity, periodicity, energy variation. Relate the observations to articulatory constriction differences.

In [10]:
```
import os
import librosa
import torch

audio_filename = "sentence2.wav"
# Sentence: "She sees you."

if not os.path.exists(audio_filename):
    print(f"Error: '{audio_filename}' not found. Please upload 'sentence2.wav' t
else:
    data, sr = librosa.load(audio_filename, sr=16000)
    waveform = torch.from_numpy(data).unsqueeze(0)
    sample_rate = 16000

    waveform = waveform / torch.max(torch.abs(waveform))
    print(f"Successfully loaded {audio_filename} at {sample_rate}Hz")
```

```
Successfully loaded sentence2.wav at 16000Hz
```

# 11. Repeat Phoneme Recognition

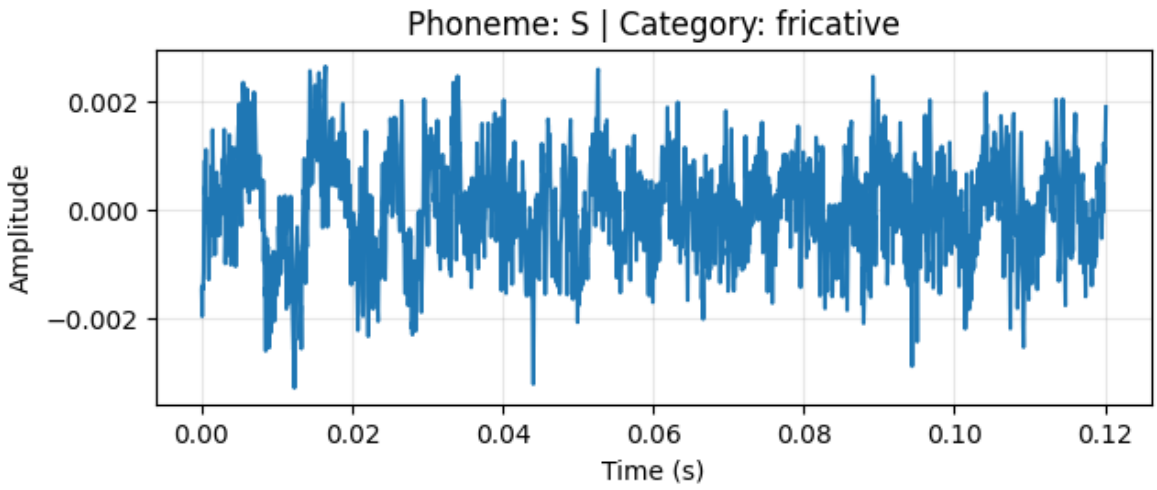
In [11]:
```
input_values = processor(
    waveform[0].numpy(),
    sampling_rate=sample_rate,
    return_tensors="pt"
).input_values

with torch.no_grad():
    logits = model(input_values).logits

predicted_ids = torch.argmax(logits, dim=-1)
```

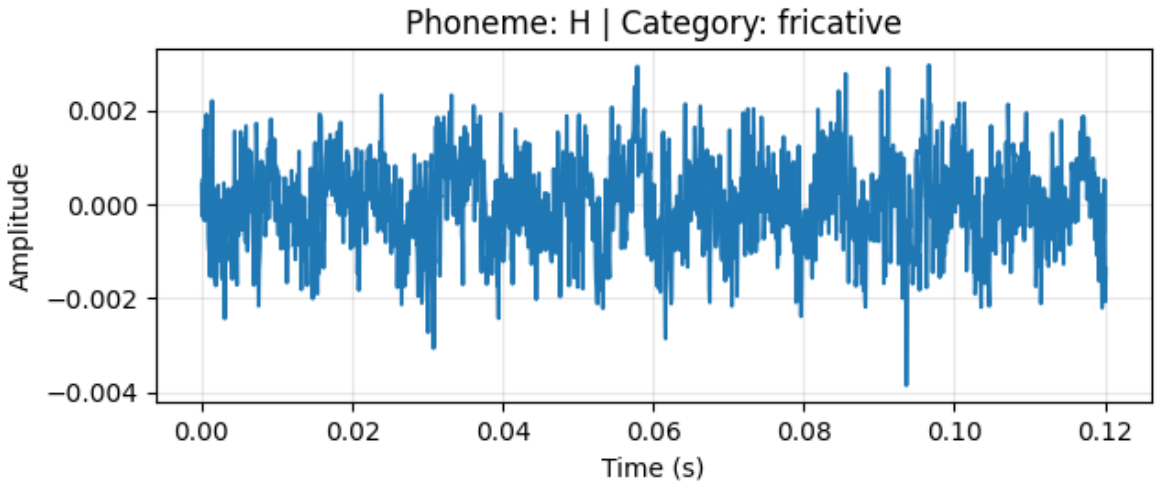
In [12]:
```
target_sampling_rate = 16000

if sample_rate != target_sampling_rate:
    waveform_np = waveform[0].numpy()
    waveform_resampled = librosa.resample(waveform_np, orig_sr=sample_rate, targ
    input_sr = target_sampling_rate
else:
    waveform_resampled = waveform[0].numpy()
    input_sr = sample_rate

input_values = processor(
    waveform_resampled,
    sampling_rate=input_sr,
    return_tensors="pt"
).input_values

with torch.no_grad():
    logits = model(input_values).logits

predicted_ids = torch.argmax(logits, dim=-1)
transcription = processor.decode(predicted_ids[0])

print("Recognized Phoneme Sequence:")
print(transcription)
```

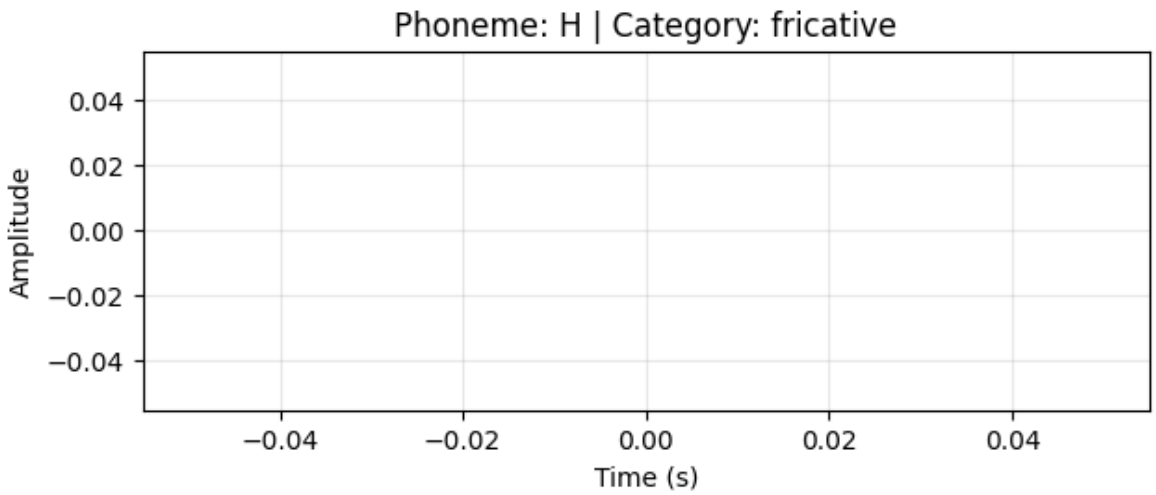
```
Recognized Phoneme Sequence:
SHE SEES YOU
```

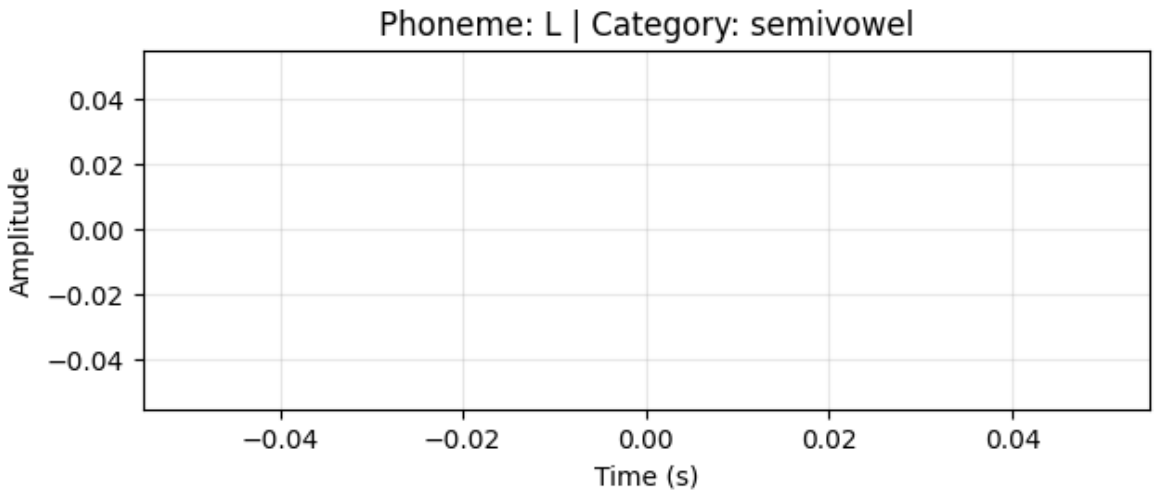
# 13. Plot Fricatives & Approximants Only

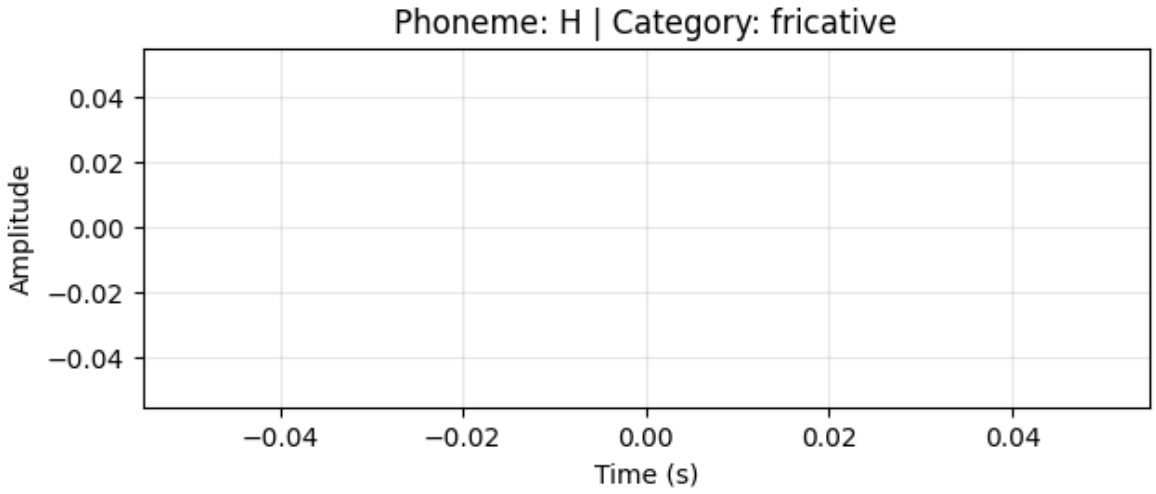
In [13]:
```
for p in phoneme_intervals:
    # Check category using uppercase match
    category = phoneme_categories.get(p["phoneme"].upper(), "other")

    if category in ["fricative", "semivowel"]:
        segment = waveform[0, p["start_sample"]:p["end_sample"]].numpy()
        plot_phoneme(segment, sample_rate, p["phoneme"], category)
```

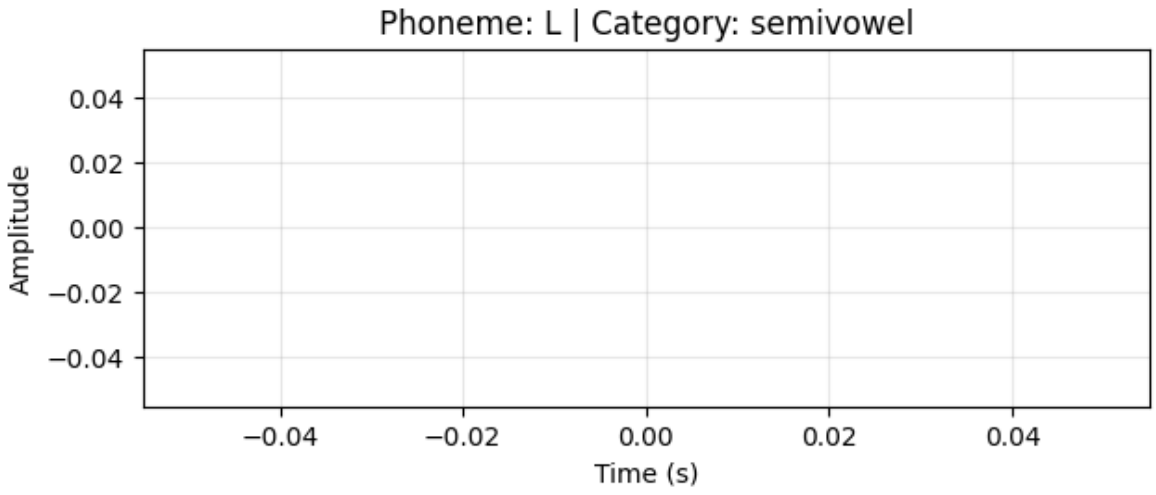

Phoneme: S | Category: fricative
Amplitude
0.002
0.000
−0.002
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Time (s)

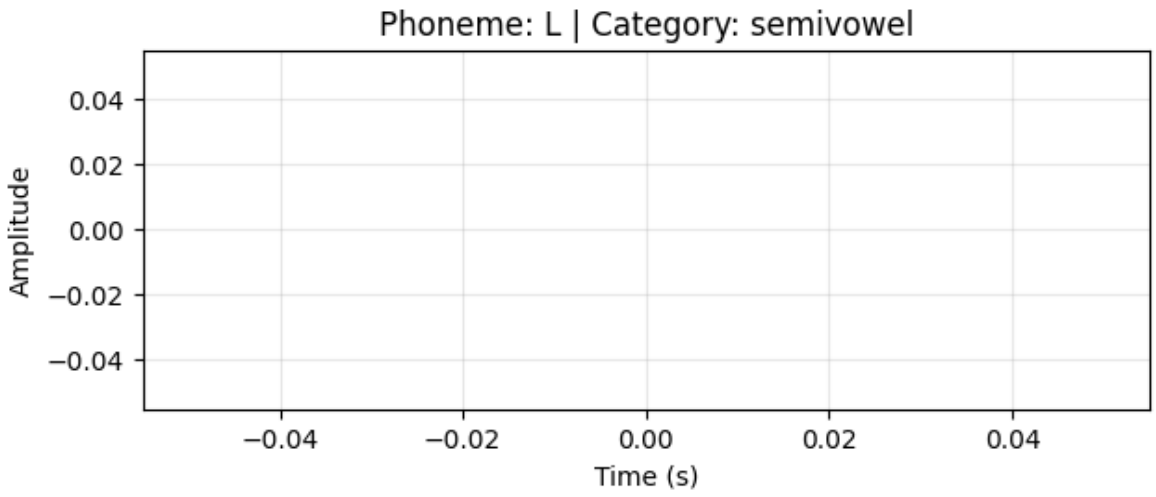

Phoneme: H | Category: fricative
Amplitude
0.002
0.000
−0.002
−0.004
0.00
0.02
0.04
0.06
0.08
0.10
0.12
Time (s)

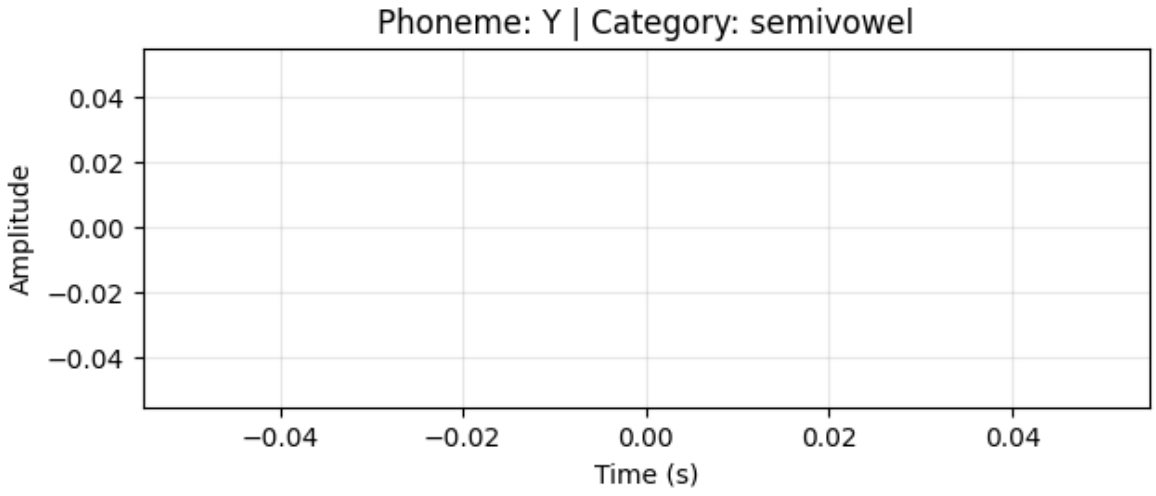

Phoneme: H | Category: fricative
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)

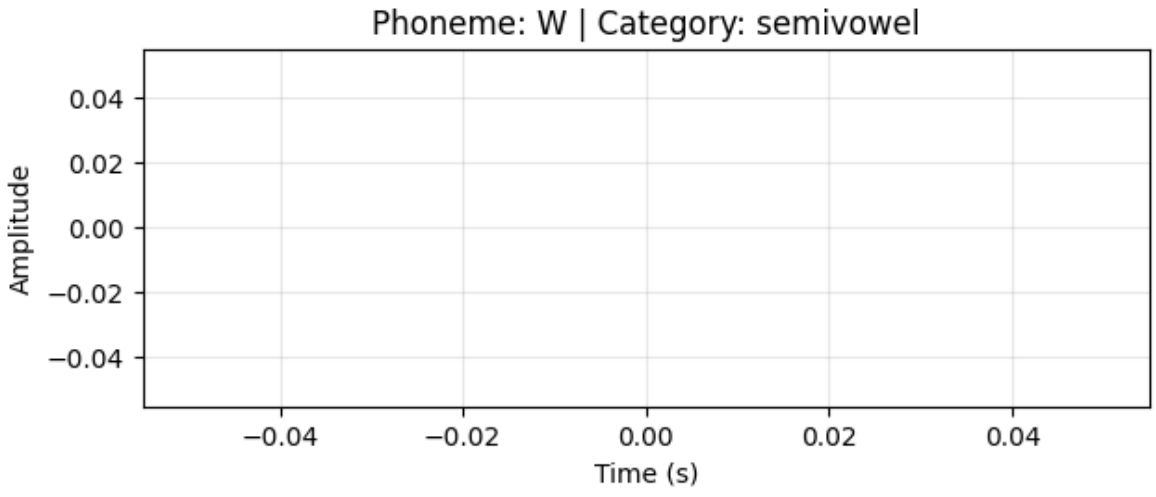

Phoneme: L | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)

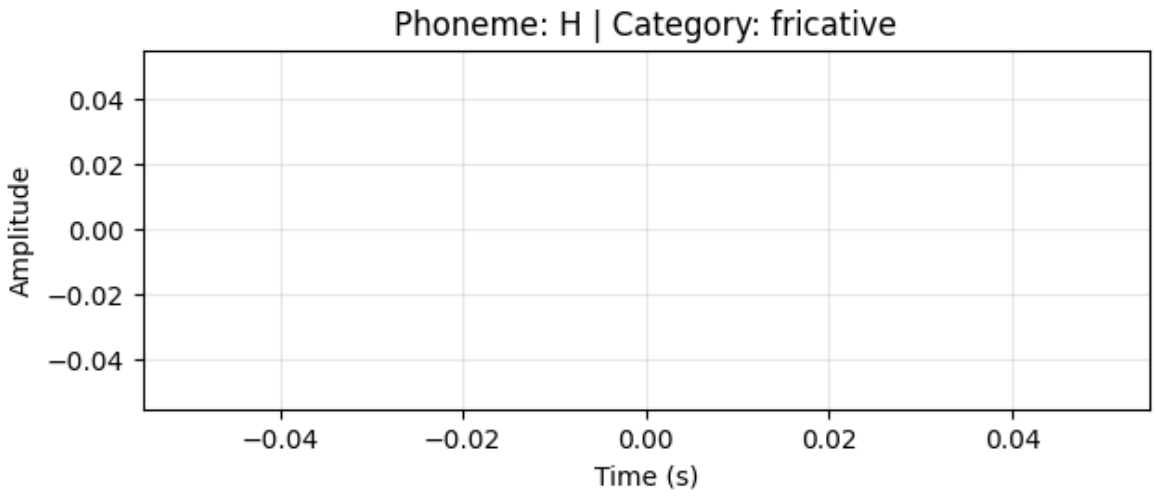

Phoneme: H | Category: fricative
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)

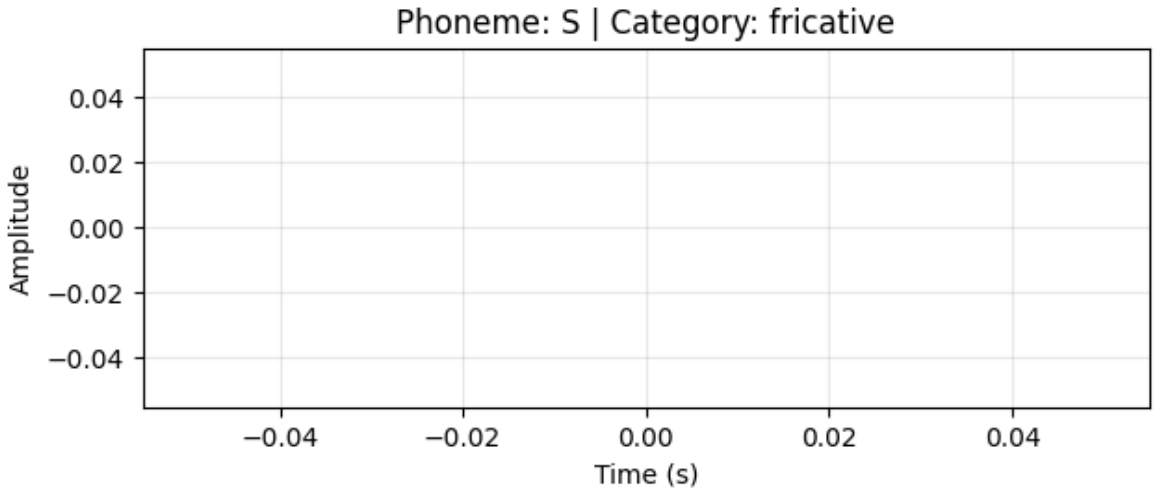

Phoneme: L | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)

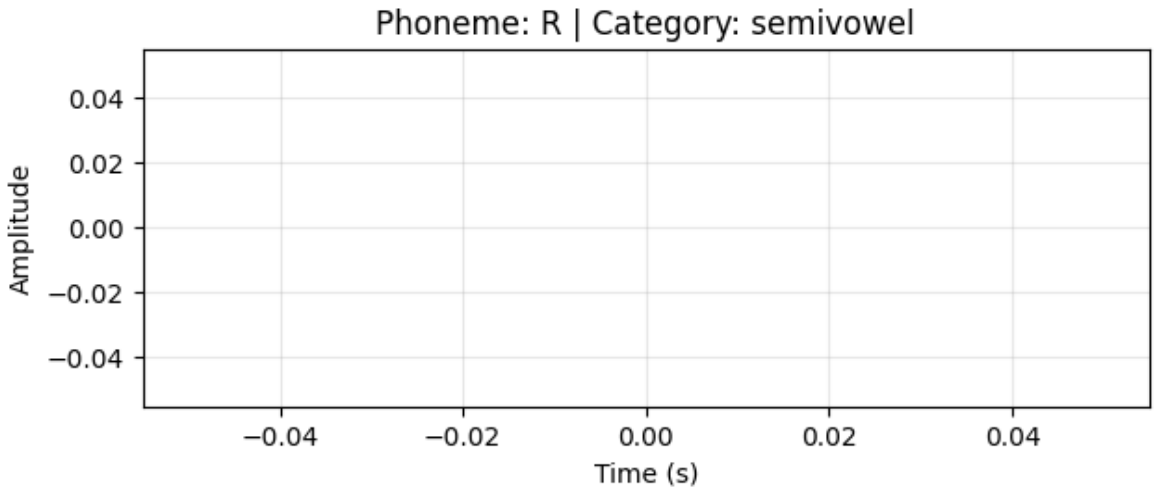

Phoneme: L | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)

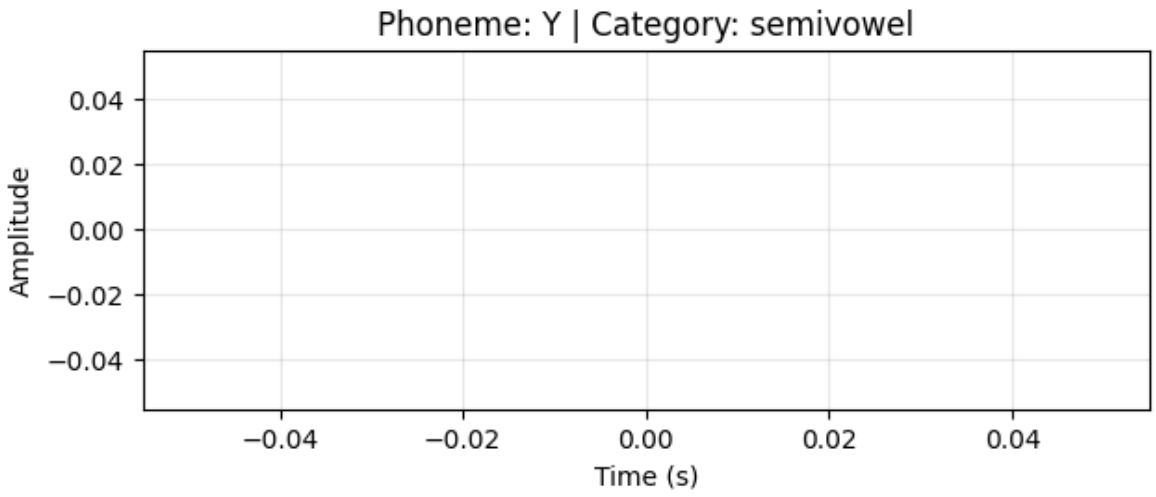

Phoneme: Y | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)

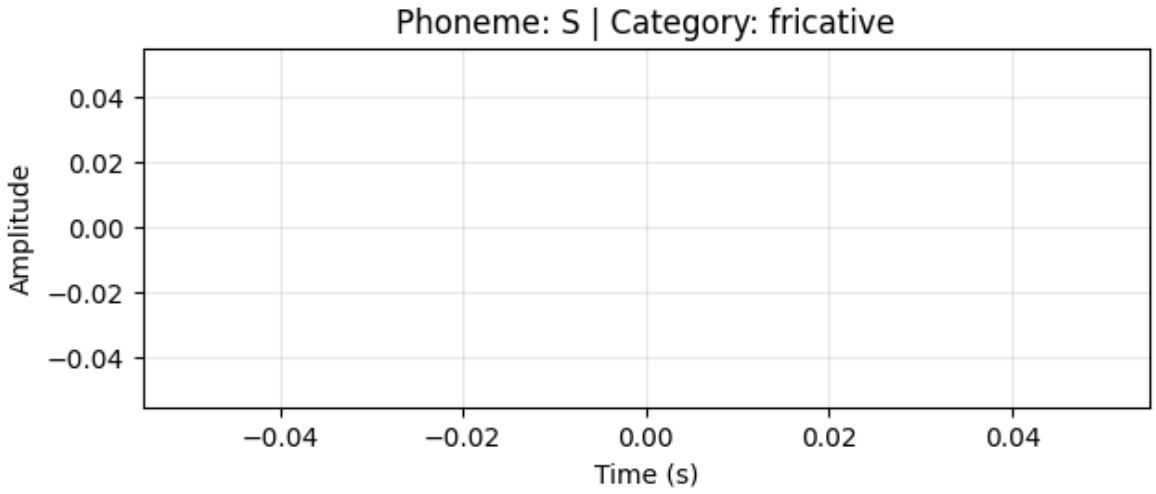

Phoneme: W | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)


Phoneme: H | Category: fricative
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)


Phoneme: S | Category: fricative
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)


Phoneme: R | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)


Phoneme: Y | Category: semivowel
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)


Phoneme: S | Category: fricative
Amplitude
0.04
0.02
0.00
−0.02
−0.04
−0.04
−0.02
0.00
0.02
0.04
Time (s)