LAB EVAL 1
UCS749: Conversational AI: Speech Processing and Synthesis

Submitted By: Pragya Gupta

Roll no: 102103407
Class: 4CO15

Submitted To: B.V. Raghav

Tasks:

1. Read and summarise the paper in about 50 words.

Ans: The Speech Commands dataset, intended for keyword spotting model training and evaluation in limited-vocabulary speech recognition, is presented in the study. It gives baseline model results, explains the dataset's features, and draws attention to the distinctions between keyword detection and generic speech recognition. Small-footprint, on-device models that are tailored for accuracy, energy efficiency, and

constrained computational resources are made possible by the dataset.

2. Download the dataset in the paper, statistically analyse and describe it, so that it may be useful for posterity. (Include code snippets in your .ipynb file to evidence your analysis.)

Ans: **1. Vocabulary:**

- The dataset contains a limited vocabulary of **35 words**. These words include:
  - Digits: "Zero" to "Nine"
  - Common commands for IoT and robotics: "Yes", "No", "Up", "Down", "Left", "Right", "On", "Off", "Stop", "Go"
  - Additional commands: "Backward", "Forward", "Follow", "Learn"
  - Auxiliary words used to evaluate non-target speech: "Bed", "Bird", "Cat", "Dog", "Happy", "House", "Marvin", "Sheila", "Tree", "Wow"

Copy of simple_audio.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment

+ Code + Text

RAM Disk

```
[12] if not os.path.exists(data_dir):
         raise FileNotFoundError(f"The directory {data_dir} does not exist.")
     else:
         print(f"Directory exists: {data_dir}")
```

```
Directory exists: /content/data
```

```
commands = np.array(tf.io.gfile.listdir(str(data_dir)))
commands = commands[(commands != 'README.md') & (commands != '.DS_Store')]
print('Commands:', commands)
```

```
Commands: ['speech_commands_v0.02.tar.gz' 'right' 'eight' 'two' 'on' 'dog' 'bed'
 'no' 'nine' 'cat' 'one' 'up' 'five' 'backward' 'left' 'learn' 'marvin'
 'go' 'follow' 'tree' 'off' 'validation_list.txt' 'testing_list.txt'
 'stop' 'zero' 'six' 'visual' 'down' 'forward' 'LICENSE' 'happy' 'house'
 'three' '_background_noise_' 'sheila' 'wow' 'seven' 'four' 'yes' 'bird']
```

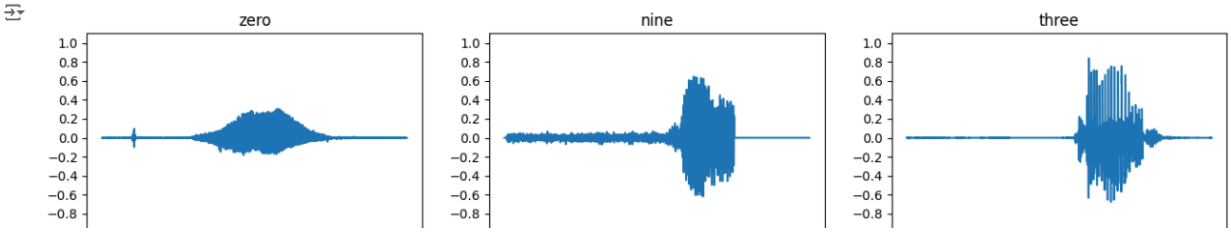
It has 36 classes:

```
train_ds, val_ds = tf.keras.utils.audio_dataset_from_directory(
    directory=data_dir,
    batch_size=64,
    validation_split=0.2,
    seed=0,
    output_sequence_length=16000,
    subset='both')

label_names = np.array(train_ds.class_names)
print()
print("label names:", label_names)
```

```
Found 105835 files belonging to 36 classes.
Using 84668 files for training.
Using 21167 files for validation.

label names: ['_background_noise_' 'backward' 'bed' 'bird' 'cat' 'dog' 'down' 'eight'
 'five' 'follow' 'forward' 'four' 'go' 'happy' 'house' 'learn' 'left'
 'marvin' 'nine' 'no' 'off' 'on' 'one' 'right' 'seven' 'sheila' 'six'
 'stop' 'three' 'tree' 'two' 'up' 'visual' 'wow' 'yes' 'zero']
```

Plotting Label name and Audio in train_ds

```
plt.figure(figsize=(16, 10))
rows = 3
cols = 3
n = rows * cols
for i in range(n):
  plt.subplot(rows, cols, i+1)
  audio_signal = example_audio[i]
  plt.plot(audio_signal)
  plt.title(label_names[example_labels[i]])
  plt.yticks(np.arange(-1.2, 1.2, 0.2))
  plt.ylim([-1.1, 1.1])
```



2. Word Utterance Distribution

- The dataset contains the following word frequency distribution (excerpt):
  - "Yes" and "Zero": 4,052 utterances each (most frequent)
  - "Backward": 1,664 utterances (least frequent)
  - Other common words such as "On", "Off", "Stop", "No", "Go" average around 3,700-4,000 utterances.

3. Speaker Diversity

- 2,618 speakers contributed to the dataset.

4. File Format

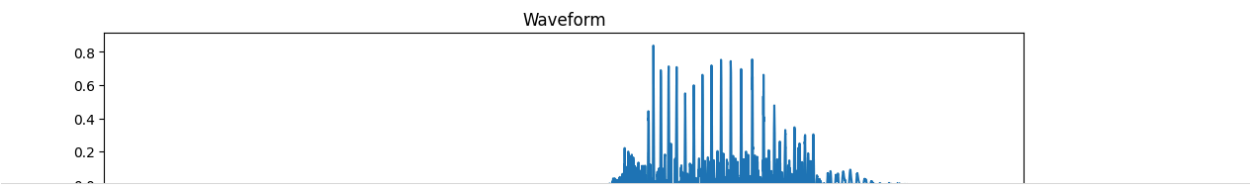
- The utterances are stored in **WAV format**, sampled at **16 KHz**, with 16-bit PCM encoding.
- Each file is **1-second long**, with zero-padding for shorter utterances.

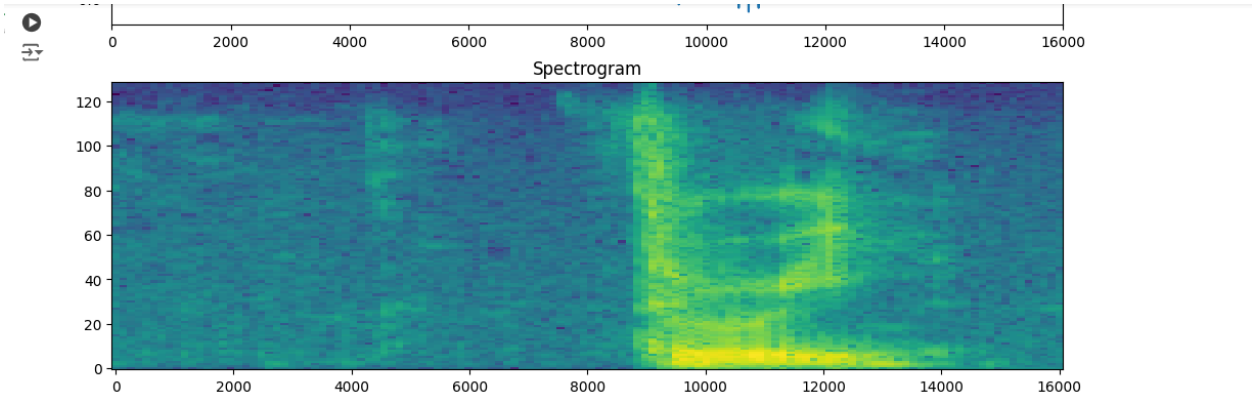
5. Noise and Background Samples

- To simulate real-world conditions, the dataset also includes **background noise** recordings (stored in a "*background_noise*" folder), which can be used to assess models' performance in distinguishing between speech and non-speech.

Plotting the example's waveform over time and the corresponding spectrogram (frequencies over time):

```
fig, axes = plt.subplots(2, figsize=(12, 8))
timescale = np.arange(waveform.shape[0])
axes[0].plot(timescale, waveform.numpy())
axes[0].set_title('Waveform')
axes[0].set_xlim([0, 16000])

plot_spectrogram(spectrogram.numpy(), axes[1])
axes[1].set_title('Spectrogram')
plt.suptitle(label.title())
plt.show()
```

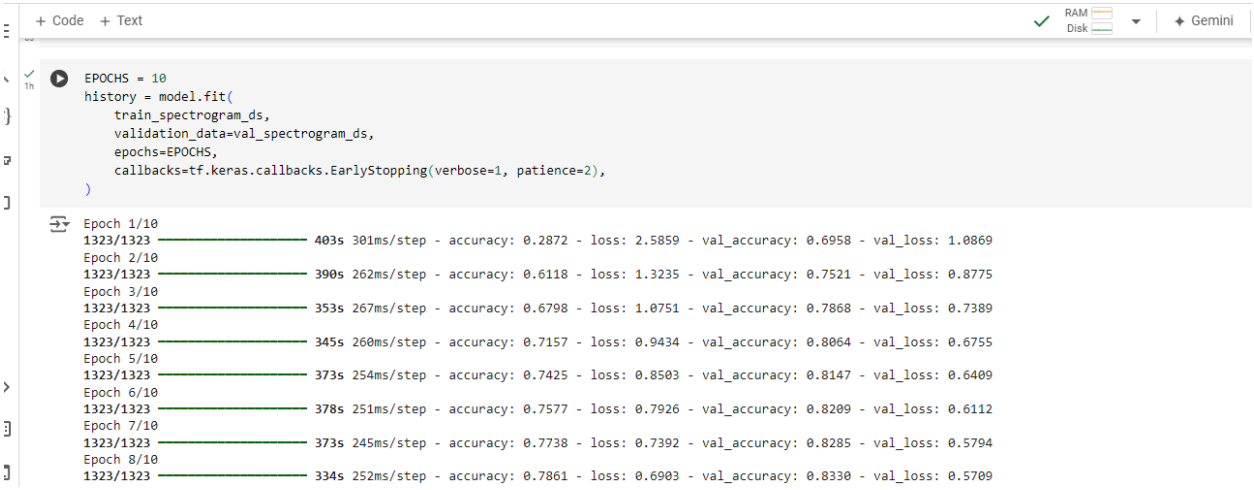




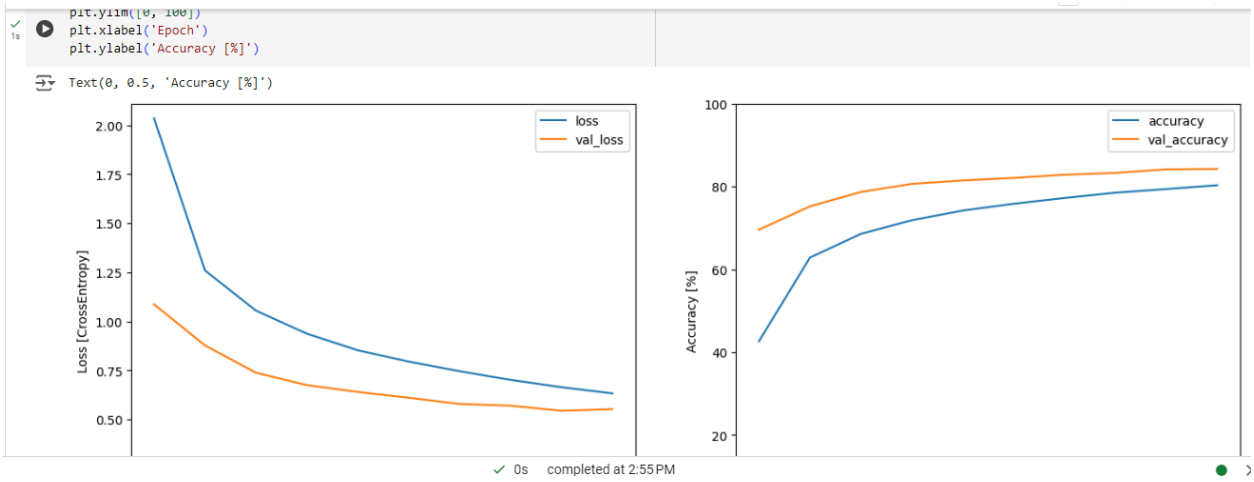
Now, create spectrogram datasets from the audio datasets:

3. Performance results using standard benchmarks.

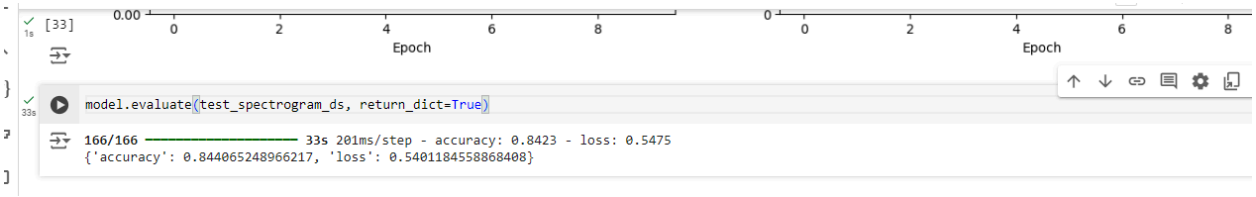
Trained for 10 epochs



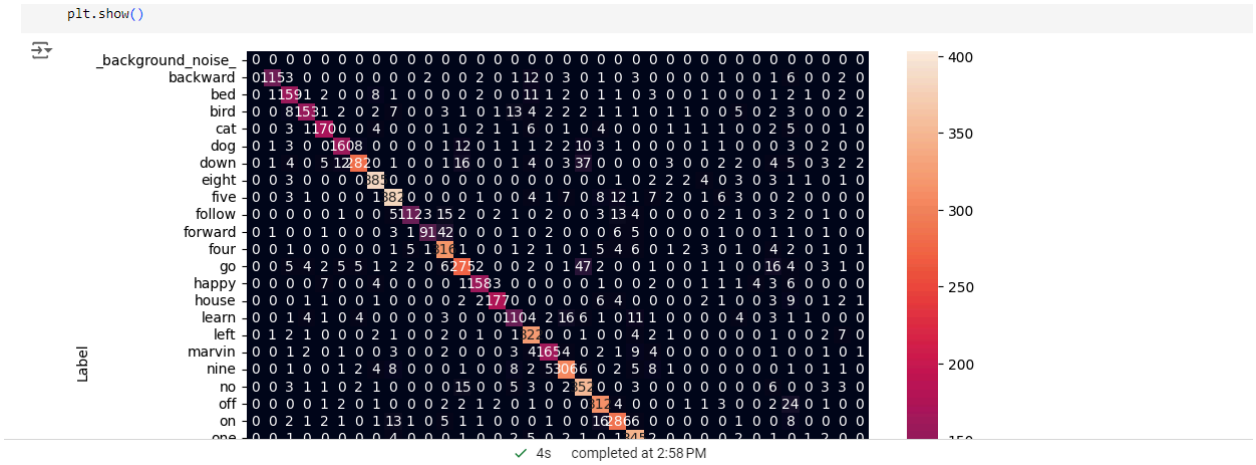
Training vs validation loss curves

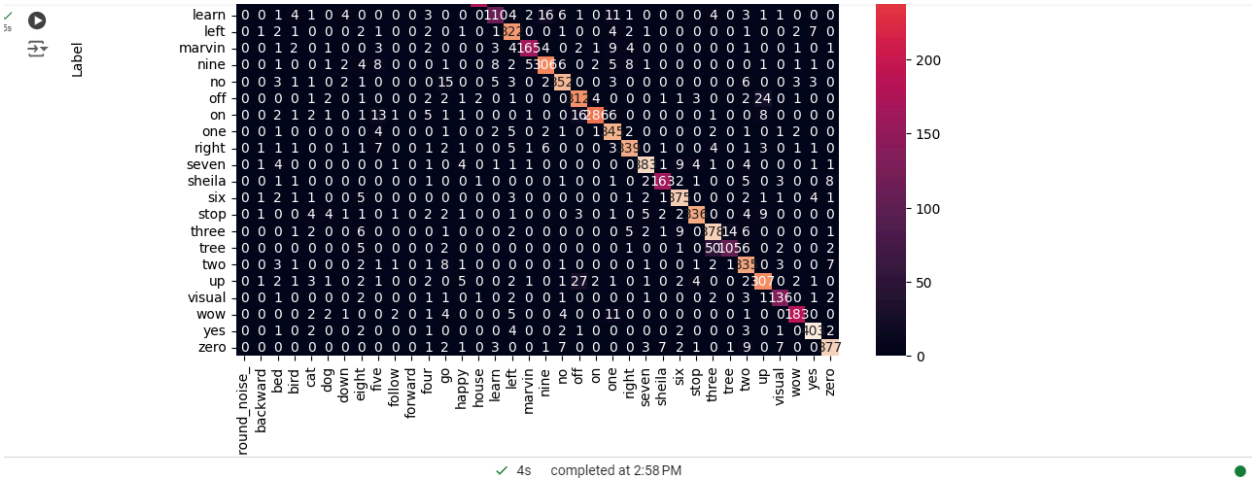


Checking the performance of the model on the test set



Confusion matrix to test how well our classifier model performs:





4. Results.

Using a recorded sample of my own voice of the word”NO”

The model predicts as follows.

The probability is the highest for the word “NO”.

```
[47] x = '/content/no.wav'
     x = tf.io.read_file(str(x))
     x, sample_rate = tf.audio.decode_wav(x, desired_channels=1, desired_samples=16000,)
     x = tf.squeeze(x, axis=-1)
     waveform = x
     x = get_spectrogram(x)
     x = x[tf.newaxis,...]

     prediction = model(x)
     x_labels = [ 'right', 'eight', 'two', 'on', 'dog', 'bed', 'no',
                  'nine', 'cat', 'one', 'up', 'five', 'backward', 'left', 'learn', 'marvin',
                  'go', 'follow', 'tree', 'off', 'stop', 'zero', 'six', 'visual',  'down', 'forward',
                  'happy', 'house', 'three', '_background_noise_', 'sheila', 'wow', 'seven', 'four', 'yes', 'bird']


     plt.bar(x_labels, tf.nn.softmax(prediction[0]))
     plt.title('No')
     plt.show()

     display.display(display.Audio(waveform, rate=16000))
```

✓ 0s completed at 3:17 PM

+ Code + Text

```
display.display(display.Audio(waveform, rate=16000))
```

1s