SPEECH LAB EVAL

NAME : KASHISH GUPTA

ROLL NO. : 102103370

SUBGROUP : 4CO14

Paper Summary :

The paper **"Speech Commands: A Dataset for Fine-Grained Speech Recognition"** presents a dataset with 65,000 one-second recordings of 12 spoken commands, such as "yes," "no," "up," and "down." It aims to advance research in speech recognition by offering a standardized benchmark for fine-grained audio classification, focusing on real-world performance in command recognition.

Google Collab files link :

<https://colab.research.google.com/drive/1dcbddNXmg2VLQ_4t2HFQZc86bKP_z6cC>

<https://colab.research.google.com/drive/1k2v-4TAlplImoT-EbDsSzEr4eQBCU640#scrollTo=SdDDteRj07ft>

<https://colab.research.google.com/drive/1auuc3uhHo1DOKkCUVxBDmp3nud7o6dVU?usp=sharing>

Analysis of data

1. A data point in the SPEECHCOMMANDS dataset is a tuple made of a waveform (the audio signal), the sample rate, the utterance (label), the ID of the speaker, the number of the utterance.

A screenshot of a computer

Description automatically generated

Number of training samples: 84843  
Number of testing samples: 11005

1. After applying fourier transformation, here is what a spectogram of label backward looks like

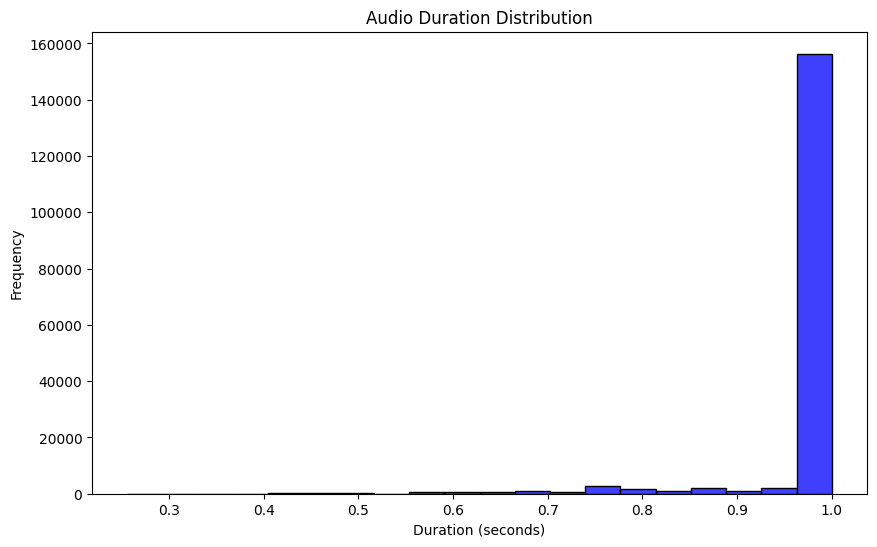
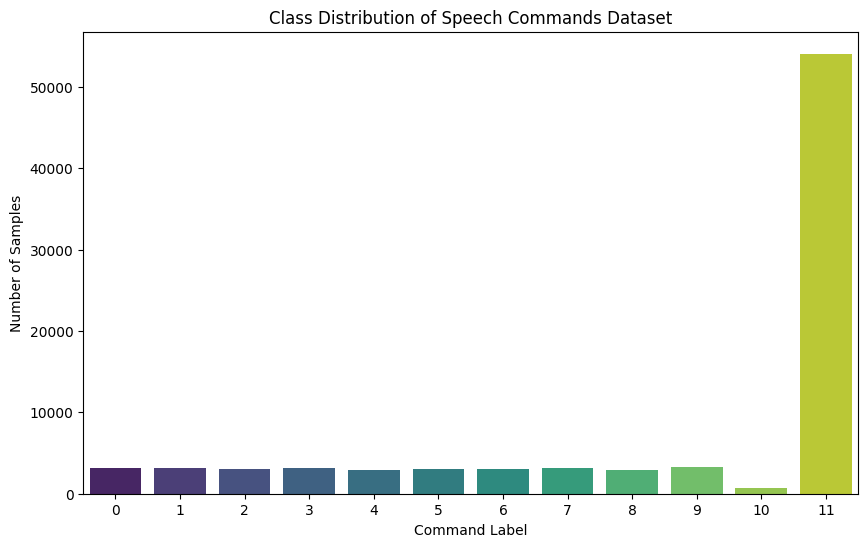
A screenshot of a computer

Description automatically generated

1. Analysis done of duration of samples in our speechcommand dataset.

A screenshot of a computer

Description automatically generated

Some snippets of analysis :A screen shot of a heat map

Description automatically generated

Steps considered while setting up data loading and batching for training and testing a model using the SPEECHCOMMANDS dataset with PyTorch.

**1. Padding Sequences (pad\_sequence function)**

* **Purpose**: Ensures all tensors in a batch are the same length by padding with zeros.
* **Details**:
  + Transposes each tensor in the batch.
  + Uses torch.nn.utils.rnn.pad\_sequence to pad tensors to the same length.
  + Permutes the dimensions of the padded tensor to match the expected input shape for your model (usually (batch\_size, channels, sequence\_length)).

**2. Collate Function (collate\_fn function)**

* **Purpose**: Defines how to collate (combine) individual data samples into a batch.
* **Details**:
  + Extracts waveforms and labels from each sample in the batch.
  + Applies pad\_sequence to ensure waveforms in the batch have the same length.
  + Converts labels to indices using label\_to\_index and stacks them into a tensor.

**3. DataLoader Configuration**

* **Purpose**: Creates DataLoader instances for the training and test datasets.
* **Details**:
  + **train\_loader**: Loads training data with batching, shuffling, and custom collate function.
    - batch\_size=256
    - shuffle=True: Shuffles data at each epoch.
    - num\_workers and pin\_memory are set based on whether a GPU (cuda) or CPU is used.
  + **test\_loader**: Loads test data with batching and custom collate function.
    - batch\_size=256
    - shuffle=False: No shuffling for test data.
    - drop\_last=False: Keeps the last batch even if it's smaller than the batch size.

Model Used (M5 Class)

**Purpose**: Defines a CNN model for classification.

**Components**:

* **Convolutional Layers**:
  + conv1: 1D convolution with kernel size 80, stride 16.
  + conv2: 1D convolution with kernel size 3.
  + conv3: 1D convolution with kernel size 3, output channels doubled.
  + conv4: 1D convolution with kernel size 3, output channels doubled.
* **Batch Normalization**:
  + bn1, bn2, bn3, bn4: Normalize outputs of respective convolutional layers.
* **Pooling Layers**:
  + pool1, pool2, pool3, pool4: Max pooling with kernel size 4.
* **Fully Connected Layer**:
  + fc1: Linear layer that maps from the output channels to the number of classes.
* **Forward Pass**:
  + Applies convolutions, batch normalization, ReLU activation, pooling, and average pooling.
  + Reshapes and passes through a fully connected layer followed by log\_softmax for classification.

A screen shot of a graph

Description automatically generated

A diagram of a company

Description automatically generated