Malayalam Text-to Speech system

Guide: Jiby J P
Project Coordinator: Manilal D.L

Group 16
Kurian Benoy 34
kurianbenoy@mec.ac.in

Objective

- To build a Text to Speech(TTS) system in Malayalam
- Obtain the state of art result

- Module1 : EDA, dataset collection
- Module2: Train first TTS system in Malayalam
- Module3: Fine tune TTS system
- Module4: User Interface

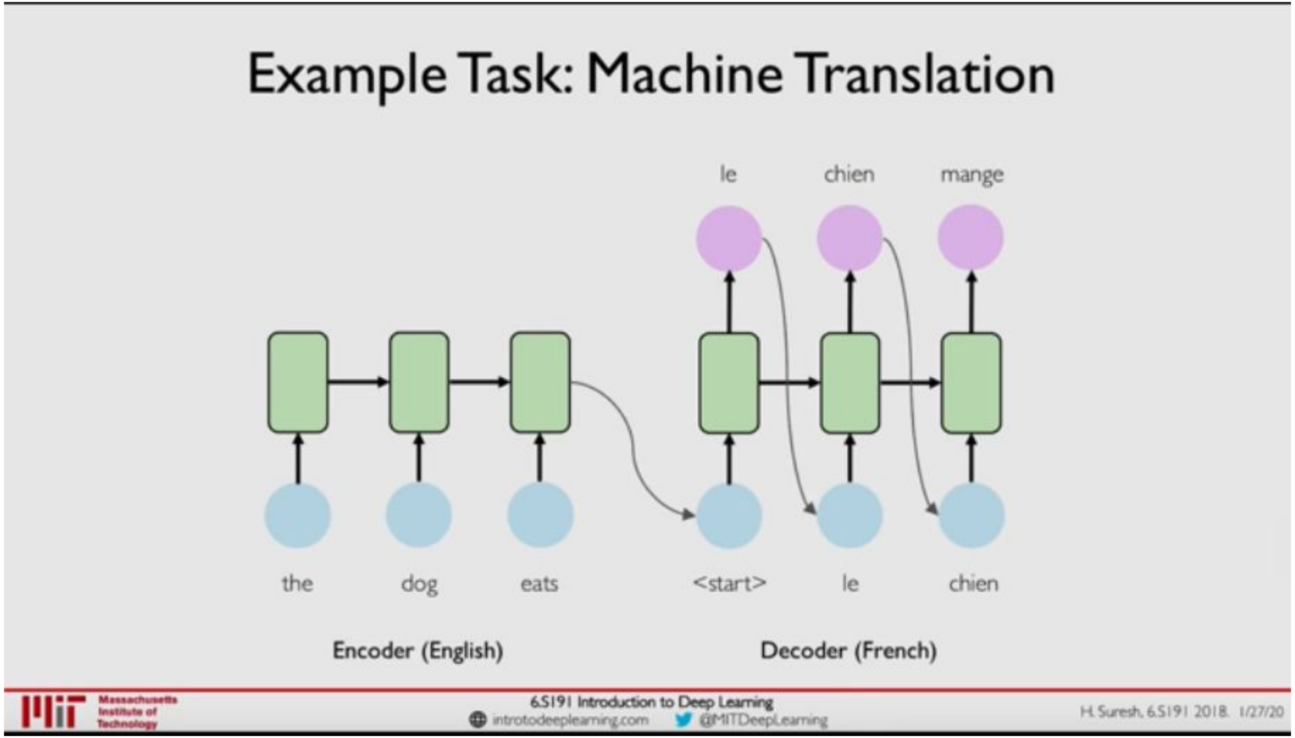
Work Done

Tactron paper Implementation

Tactron model is a sequence to sequence model, which is Currently one of the best Text to speech architectures. An improved architecture called Tactron2 also exists. Seq-2-seq models use RNNs as backbone

Aim: To understand the paper and implement architecture

Sequence to Sequence models



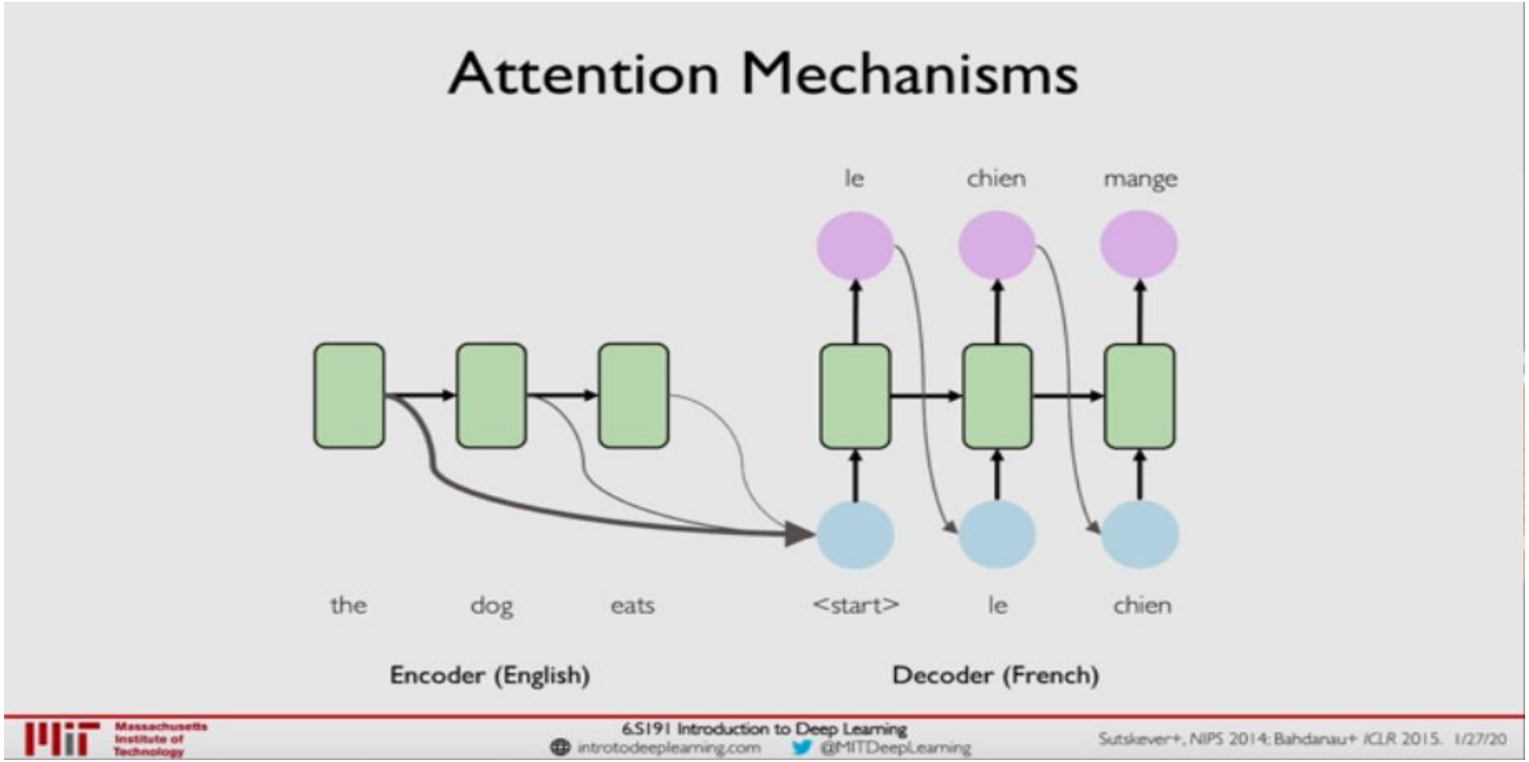
Example Task: Machine Translation

Massachusetts Institute of Technology

6.S191 Introduction to Deep Learning
introtodeeplearning.com @MITDeepLearning

H. Suresh, 6.S191 2018. 1/27/20

Sequence to Sequence models



Attention Mechanisms

Massachusetts Institute of Technology

6.S191 Introduction to Deep Learning

introtodeeplearning.com @MITDeepLearning

Sutskever+, *NIPS* 2014; Bahdanau+ *ICLR* 2015. 1/27/20

Components of Tactron architecture

1. CBHG module
2. Encoder
3. Decoder

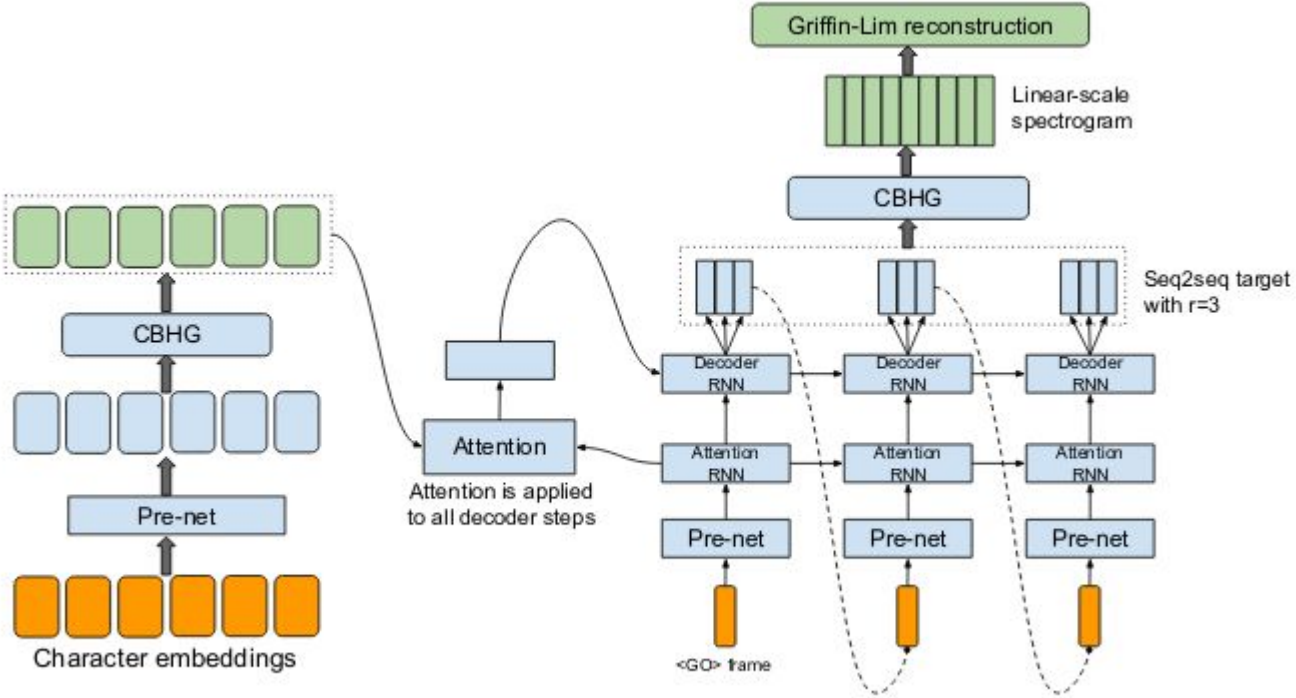
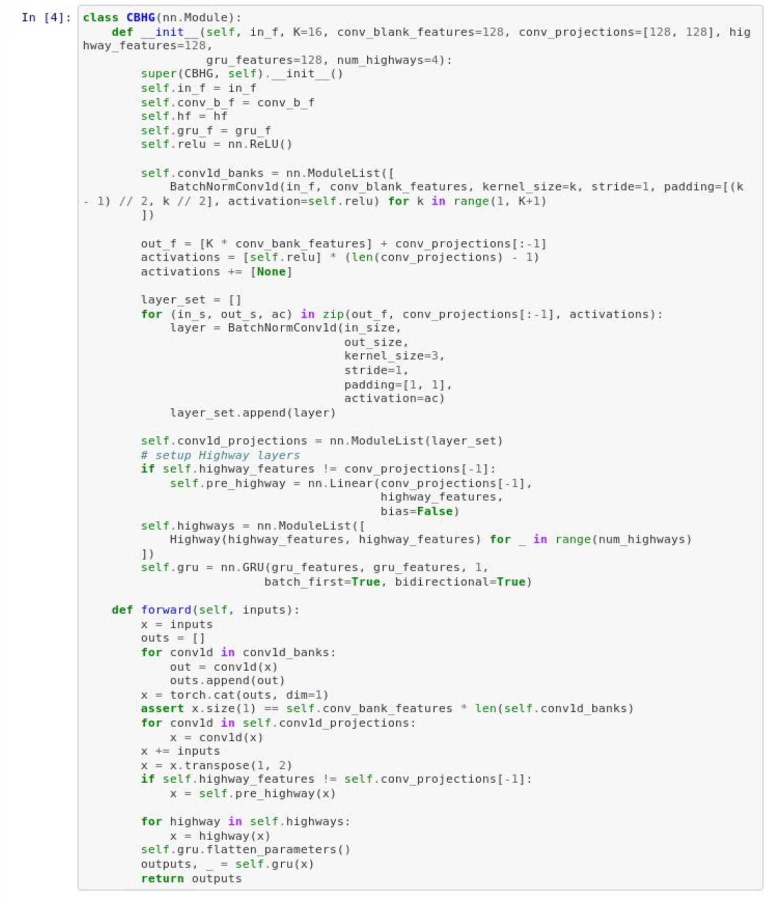


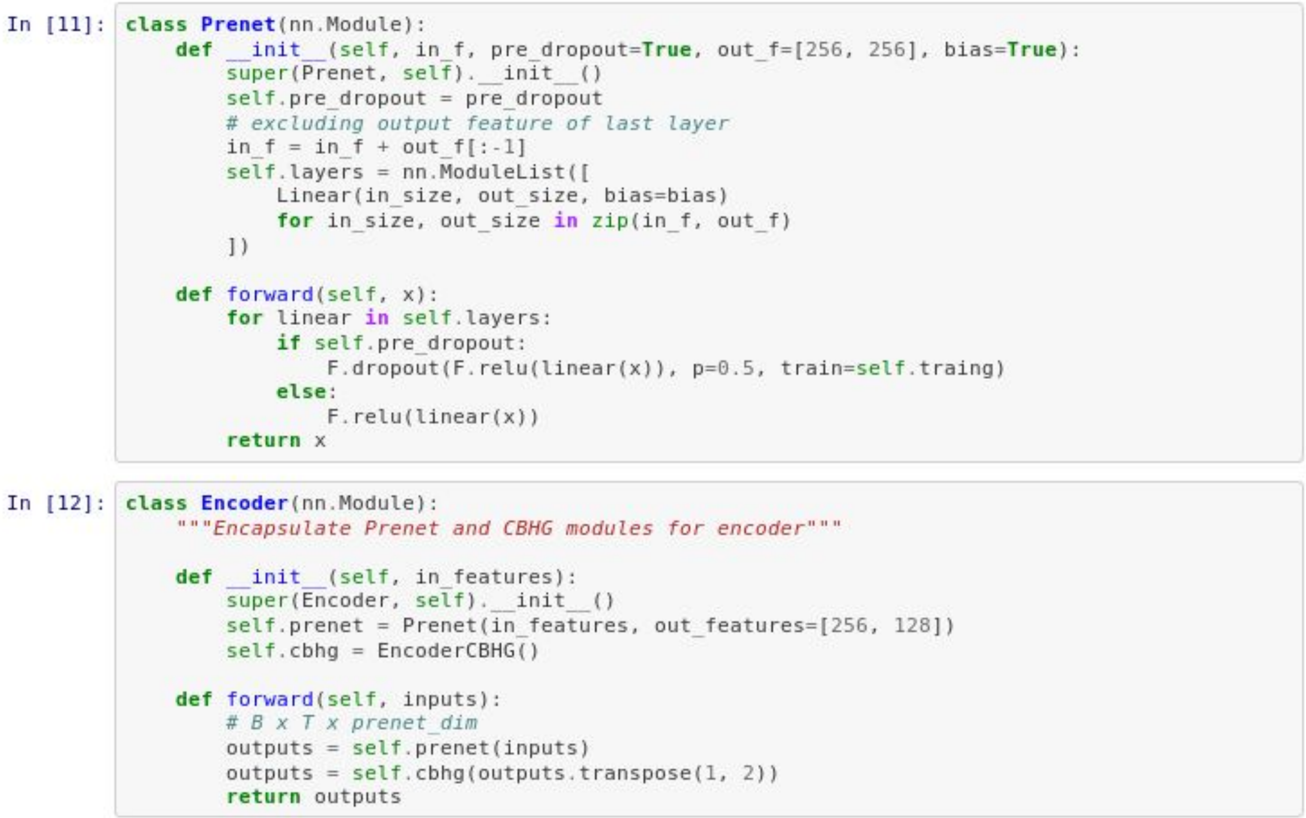
Figure 1: *Model architecture. The model takes characters as input and outputs the corresponding raw spectrogram, which is then fed to the Griffin-Lim reconstruction algorithm to synthesize speech.*

CBHG Module



```
In [4]: class CBHG(nn.Module):
            def __init__(self, in_f, K=16, conv_blank_features=128, conv_projections=[128, 128], hig
        hway_features=128,
                             gru_features=128, num_highways=4):
                super(CBHG, self).__init__()
                self.in_f = in_f
                self.conv_b_f = conv_b_f
                self.hf = hf
                self.gru_f = gru_f
                self.relu = nn.ReLU()

                self.conv1d_banks = nn.ModuleList([
                    BatchNormConv1d(in_f, conv_blank_features, kernel_size=k, stride=1, padding=[(k
        - 1) // 2, k // 2], activation=self.relu) for k in range(1, K+1)
                ])

                out_f = [K * conv_bank_features] + conv_projections[:-1]
                activations = [self.relu] * (len(conv_projections) - 1)
                activations += [None]

                layer_set = []
                for (in_s, out_s, ac) in zip(out_f, conv_projections[:-1], activations):
                    layer = BatchNormConv1d(in_size,
                                            out_size,
                                            kernel_size=3,
                                            stride=1,
                                            padding=[1, 1],
                                            activation=ac)
                    layer_set.append(layer)

                self.conv1d_projections = nn.ModuleList(layer_set)
                # setup Highway layers
                if self.highway_features != conv_projections[-1]:
                    self.pre_highway = nn.Linear(conv_projections[-1],
                                                 highway_features,
                                                 bias=False)
                self.highways = nn.ModuleList([
                    Highway(highway_features, highway_features) for _ in range(num_highways)
                ])
                self.gru = nn.GRU(gru_features, gru_features, 1,
                                  batch_first=True, bidirectional=True)

            def forward(self, inputs):
                x = inputs
                outs = []
                for conv1d in conv1d_banks:
                    out = conv1d(x)
                    outs.append(out)
                x = torch.cat(outs, dim=1)
                assert x.size(1) == self.conv_bank_features * len(self.conv1d_banks)
                for conv1d in self.conv1d_projections:
                    x = conv1d(x)
                x += inputs
                x = x.transpose(1, 2)
                if self.highway_features != self.conv_projections[-1]:
                    x = self.pre_highway(x)

                for highway in self.highways:
                    x = highway(x)
                self.gru.flatten_parameters()
                outputs, _ = self.gru(x)
                return outputs
```

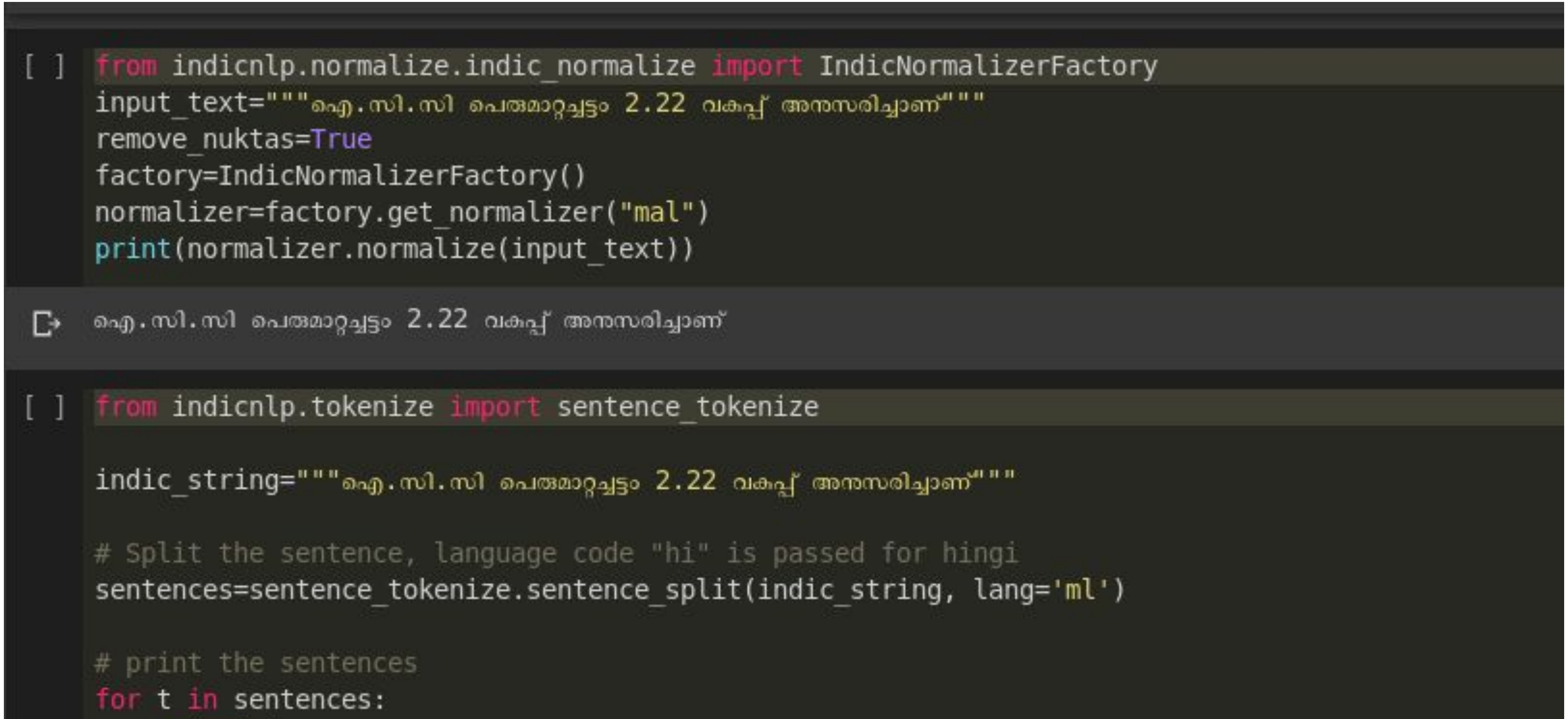
Encoder Module



```
In [11]: class Prenet(nn.Module):
             def __init__(self, in_f, pre_dropout=True, out_f=[256, 256], bias=True):
                 super(Prenet, self).__init__()
                 self.pre_dropout = pre_dropout
                 # excluding output feature of last layer
                 in_f = in_f + out_f[:-1]
                 self.layers = nn.ModuleList([
                     Linear(in_size, out_size, bias=bias)
                     for in_size, out_size in zip(in_f, out_f)
                 ])

             def forward(self, x):
                 for linear in self.layers:
                     if self.pre_dropout:
                         F.dropout(F.relu(linear(x)), p=0.5, train=self.traing)
                     else:
                         F.relu(linear(x))
                 return x
```

```
In [12]: class Encoder(nn.Module):
             """Encapsulate Prenet and CBHG modules for encoder"""

             def __init__(self, in_features):
                 super(Encoder, self).__init__()
                 self.prenet = Prenet(in_features, out_features=[256, 128])
                 self.cbhg = EncoderCBHG()

             def forward(self, inputs):
                 # B x T x prenet_dim
                 outputs = self.prenet(inputs)
                 outputs = self.cbhg(outputs.transpose(1, 2))
                 return outputs
```

Text normalisation



```
from indicnlp.normalize.indic_normalize import IndicNormalizerFactory
input_text="""ഐ.സി.സി പെരുമാറ്റച്ചട്ടം 2.22 വകുപ്പ് അനുസരിച്ചാണ്"""
remove_nuktas=True
factory=IndicNormalizerFactory()
normalizer=factory.get_normalizer("mal")
print(normalizer.normalize(input_text))
```

```
ഐ.സി.സി പെരുമാറ്റച്ചട്ടം 2.22 വകുപ്പ് അനുസരിച്ചാണ്
```

```
from indicnlp.tokenize import sentence_tokenize

indic_string="""ഐ.സി.സി പെരുമാറ്റച്ചട്ടം 2.22 വകുപ്പ് അനുസരിച്ചാണ്"""

# Split the sentence, language code "hi" is passed for hingi
sentences=sentence_tokenize.sentence_split(indic_string, lang='ml')

# print the sentences
for t in sentences:
```

Work on top of Mozilla TTS to train first notebook

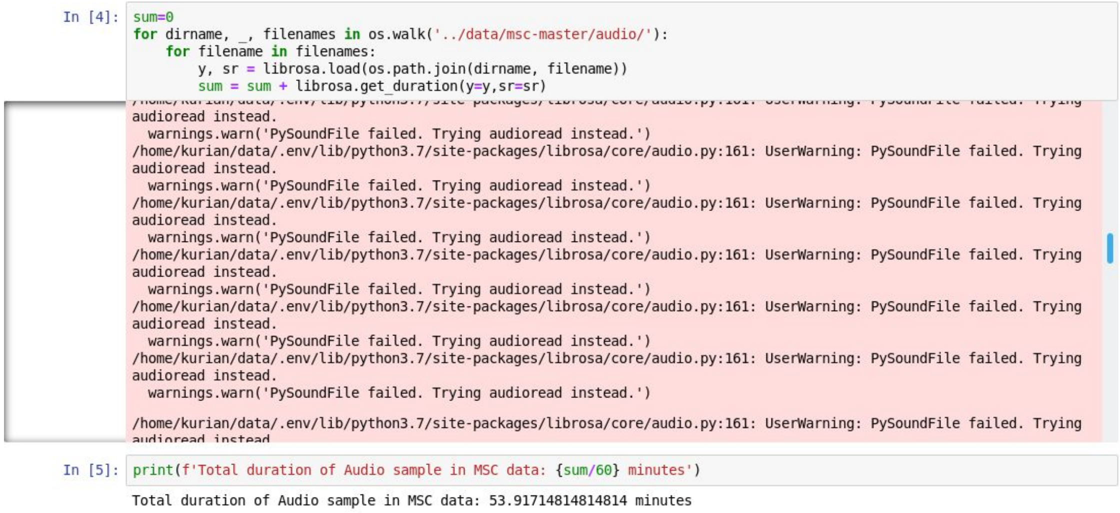
- Collect data
- Do dataset preprocessing
- Text normalisation
- Initial training pipeline

Training Notebook

https://drive.google.com/drive/u/2/folders/1gZbGSVbVnaX87XO9E-nGYxJcZC4f12rg

EDA of Malayalam Speech corpora

Total duration of dataset



```
In [4]: sum=0
        for dirname, _, filenames in os.walk('../data/msc-master/audio/'):
            for filename in filenames:
                y, sr = librosa.load(os.path.join(dirname, filename))
                sum = sum + librosa.get_duration(y=y,sr=sr)
```

```
/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead.
  warnings.warn('PySoundFile failed. Trying audioread instead.')
/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead.
  warnings.warn('PySoundFile failed. Trying audioread instead.')
/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead.
  warnings.warn('PySoundFile failed. Trying audioread instead.')
/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead.
  warnings.warn('PySoundFile failed. Trying audioread instead.')
/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead.
  warnings.warn('PySoundFile failed. Trying audioread instead.')
/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead.
  warnings.warn('PySoundFile failed. Trying audioread instead.')

/home/kurian/data/.env/lib/python3.7/site-packages/librosa/core/audio.py:161: UserWarning: PySoundFile failed. Trying
audioread instead
```

```
In [5]: print(f'Total duration of Audio sample in MSC data: {sum/60} minutes')
```

```
Total duration of Audio sample in MSC data: 53.91714814814814 minutes
```

Research paper

A study on Text to speech systems for Non-English languages

(featured in IJRAR Research Journal)
http://www.ijrar.org/papers/IJRAR19K8100.pdf

Thank you!