Esther Oduntan AMMI_2020
The github Link is:
https://github.com/estheroduntan/AMMI_2020_SPEECH_RECOGNITION_PROJECT
Speech Recognition Project Report:

Introduction:
In completion of the course on speech recognition; a project with the tiltle: Performing a Text to Speech Task using customized audio data. A Case study of Yoruba Language; was carried out.
Speech Recognition is is an interdisciplinary subfield of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language into text by computers. There are three categories of these concepts namely: Text To Speech(TTS); Automatic Speech Recognition(ASR) and Speech To Text(STT).

This project can be categorised as Text To Speech(TTS). In other to perform the TTS task, the following steps were taken:

I. Creating the Dataset
1. transcribing of text into an electronic format of .txt; the .txt file were organised into sessions, each session comprises of minimum of fifty(50) senetences.
2. Downloading and installing of the LIG Aikuma API on an Android phone, to enable the transcription of the .txt file to audio format of .wav.
3. On the LIG Aikuma API, the elicitation by Text was performed on each session of .txt file; which serves as input and the output were .json, .wav and linker file.
4. The output from the Android API was transferred to the laptop for preprocessing.
5. the audio files(.wav) were timed using vlc application and it reported a total of 2hrs 26 seconds.

II. Text Preprocessing:
Text preprocessing is an act of cleaning and preparing text data. In this project, the pre-processing was performed by developing a python script named: generate.py. This script was used to define the set of characters used in the yoruba language; it was also used to remove corrupted files in the data set, it was used to derive the all_session.txt file which was used for the model implementation. The all_session.txt file was tokenized and encoded into numeric format.

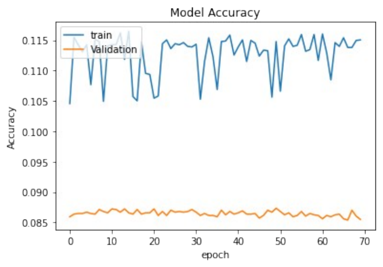
The dataset was splitted into: train, validation and test using the ratio of 40%, 20%, and 1 hour respectively.

III. Model Building and Training:
In other to train the model, python libraries such as torchaudio, PyDrive, soundfile were installed and GoogleAuth, GoogleDrive, GoogleCredentials were imported using Python programming language in Colab environment. The task performed at this stage:

a. **Contrastive Predictive Coding(CPC)**: The raw audio wave was paased through the convolutional network encoder; then, the encoder’s output was given to a recurrent network that is the context. The prediction network was used to predict the furure embeddings of the encoder using the output of the context network.
b. **Generating the CPC Loss**: The CPCCriterion was used to hold the prediction networks and the classification loss L was derived. The training loop was implemented using Adam optimizer, the dataloader provided by the CPC_audio library was used. An average Loss of 4.87 was derived at the end of the training loop.
c. **The model parameters were Fine tunned by:**
1. Performing Phone separability with aligned phonemes that is using CPC to recognise phonemes. An evaluation setting was trained in which there is phone labels for each timestep. A simple linear classifer was used to recognise the phonemes from the features prodiuced by cpc_model. Parameter: learning rate used: 2e-4 and Adam optimizer. The output of the Fine tuning gave an average loss of

3.445 and average accuracy of 0.0855 for validation dataset and 3.270 and 0.115 for average loss and average accuracy.

The training loop was implemented for 70 epochs. At the end of the 10th epoch we had:



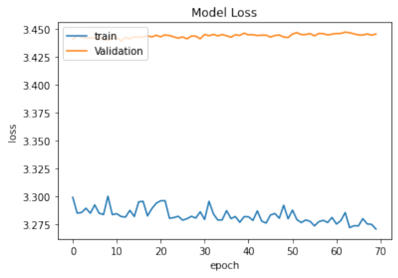


Figure 2:Model Accuracy Figure 3: Model Loss

Running epoch 70 / 70 -------------------
Training dataset : Average loss : 3.270360166376287. Average accuracy 0.11505681818181818
Validation dataset Average loss : 3.4450841744740806. Average accuracy 0.08550347222222222

2. **Performing Phone separabilty without alignment(PER)**
In this case, the functions: tain_one_epoch_ctc, validation_step_ctc and run_ctc was implemented. The average loss for the training and validation dataset derived at the completion of 70 epochs is as follows:

Running epoch 70 / 70 -------------------
Training dataset : Average loss : 2.639800259045192.
Validation dataset Average loss : 2.429150436863755

The PER without alignment evaluation of the validation loss and training loss of 70 epochs is represented with the plot below:

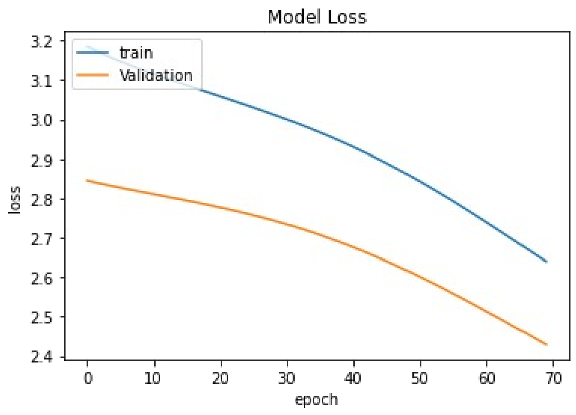


Figure 3:PER Model Loss for Train and Validation Dataset
The PER for the train data is: 0.9645712432332751

3. **Performing Character Error Rate(CER)**
This involves the use of Characters instead of Phonemes in evaluating the performance of the cpc-model. The Avaerage loss on the training data set and the validation dataset after 50 epochs are as follows:

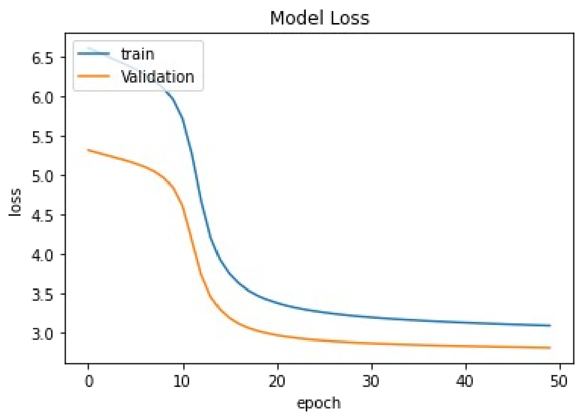


Figure4:CER Model Loss for Train and Validation Dataset

The Average CER computed on the train and validation data through beam search gave: 0.9647507064867658.

Performing Character Error Rate(CER) and PER on the Test Data
The result shows: 57.92342491328996 and 0.9844334771946059 respectively

Problems Encountered:
The following problems were encountered:
1. **Corrupted .wav files:** some .wav files were observed to be corrupted duing the preprocessing stage, which made it difficult for such files to the read by the scripts.
Solution: This was addressed using the preprcoessing code(generate.py), the corrupted files were scanned out and removed.
2. **Unformatted text/sentences:** it was observed that some sentences in the session were not preceeded with "##", which made it difficult for the Li….API to read the text file as a .txt file.
Solution: the .txt files with these omission were identified and amended.
3. **Uncaptured Character:** During the preprocessing, it was observed that some characters were not captured in the list of alphabets allowed in the yoruba language; these uncaptured characters were reported as "Key Error"
Solution: generate.py script which was written was used to fix the problem and the set of alphabets were updated.

Knowledge Gained:
In this project, I have learnt Text to Speech practically; which is very encouraging; also I have acquired skills as follows:
- Creation of .wav audio dataset from text
- Training and use of cpc-model and ctccriterion loss;
- Ability to fine tune audio dataset to reduce training and validation loss

State what directions you did not have time to address and which ones seem most promising:
The direction that could not be addressed is the ability to fine tune with some other languages due to time and infrastructures. Inabibility to record more audio file due to time constraints.
The most promising is the data set creation;that is the transcription and recording; which was tasking and interesting.

Conclusion:
In this project, I have been able to carried out a Text To Speech Task,train the model using cpc, optimze the model using Adams optimzer, Fine tune the model by adjusting parameters such the epochs, learning rates, applying alignment on PER and removal of alignment. Also, Character Error Rate(CER) was used on all data sets(Train, Validation and Test).