AI/ML SPEECH RECOGNITION PROJECT

N SAI VENKATA SARATH CHANDRA CC20MTECH14001

Guided by:
Dr G.V.V.SHARMA

28-12-2020

INTRODUCTION

1. Speech recognition 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.
2. This Speech command recognition model is based on concepts of Convolution, LSTM and Attention.

CREATE DATA

1. Setting 16KHz as sampling rate
2. Recording 80 utterances of each command.
3. Trimming each utterance to one second.
4. Saving samples of each command in different folders
   Dataset/forward
   Dataset/back
   Dataset/left
   Dataset/right
   Dataset/stop
5. Using Audacity to record the voice samples for words forward,back,left,right,stop.

LOAD DATA

1. Load Data using numpy files.
2. Wavefile, librosa etc can also be used.

SPLIT DATASET

Carrying out a stratified split of the dataset into train and test set with 20Set a random seed for reproducing the split.

AUGMENT DATA

Augment each audio sample by time shifting in 25000 length vectors filled with zeros.
Take steps of 500 to create 18 files per sample

FEATURE EXTRACTION

1. MFCCs are most prominent features used in audio processing.
2. Normalizing the MFCCs over the frequency axis is found to reduce effect of noise.
3. Kapre is a python package that provides layers for audio processing that are compatible with keras and utilize GPU for faster processing. Kapre provides us with a layer basically

BUILDING MODEL: DESCRIPTION

1. Using convolutional layers ahead of LSTM is shown to improve performance in several research papers.
2. Batch normalization layers are added to improve convergence rate.
3. Using Bidirectional LSTM is optimal when complete input is available. But this increases the runtime two-fold.
4. Final output sequence of LSTM layer is used to calculate importance of units in LSTM using a FC layer.
5. Then take the dot product of unit importance and output sequences of LSTM to get attention scores of each time step.
6. Take the dot product of Attention scores and the output sequences of LSTM to get attention vector.
7. Add an additional FC Layer and then to output Layer with SoftMax Activation.

RUN

- The Code is written using GOOGLE COLAB.
- Open Colabnotebook.ipynb and changing Runtime to GPU.
- Upload file to Colab.
- Change datadir in all cells to point to Dataset.
- Run the cells in order in the notebook.

TESTING

1. Augment the test set same as training set.
2. Extract MFCCs using same method as training set
3. Test set is passed as validation set to fit method of model.
4. The performance of model on test set is calculated after every epoch.

TEST STEPS

1. Locating the model.h5 file.
2. Start speaking when mike is seen at the bottom right of the task bar.

VISUALIZE ATTENTION

1. Building a sub model from the trained model.
2. Taking same input layer and adding 'AttentionSoftmax' layer as additional output layer.
3. Passing MFCCs of test samples to predict method.
4. Now plotting log of attention Scores and corresponding input vector before taking MFCCs on different axes.

REFERENCES

1. A neural attention model for speech command recognition - by Douglas Coimbra de Andrade, Sabato Leo, Martin Loesener Da Silva Viana, Christoph Bernkopf
2. GITHUB:https://github.com/PradeepMoturi/Speech-Command-Model/tree/master/Data/Pradeep16
3. GITHUB:https://github.com/gadepall/aiml/blob/master/Speech-Command-Model/Report.pdf