Predicting Spoken Words Through A Neural Network

Nicholas Dowley

Acknowledgements

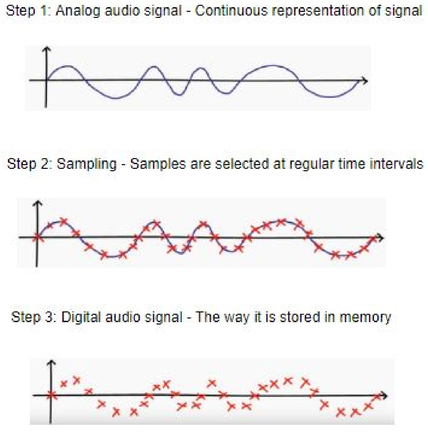
Text Box

Introduction

In today's everyday life, fewer and fewer people are utilizing their keyboards to search for something online. Instead, they opt to use their voice, asking their phones, Amazon Alexa, or Google Home to search the web for them. These devices are able to comprehend the spoken language, print out or repeat what was asked, and perform the action. But how does a machine understand what we say? The answer is through a Deep Learning. Through Deep Learning, machines utilize a neural network that simulates neurons in the human brain to train and process inputs so that it will be able to make predictions when test inputs are added.

Methods

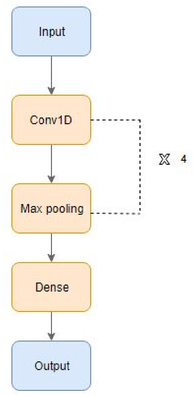
The dataset used for this was a set of multiple one-second words taken from Kaggle. Using Python through Jupyter Notebook, the dataset was loaded. The data consisted of ten spoken words. When using audio based data, it is necessary to implement the process of sampling, which converts analog audio signals into digital audio signals.



Once the audio files were sampled, they were integer encoded, then reshaped into a 3D array to prepare for the neural network.

To teach the program, a conv1d model was used. This is a type of neural network that performs along only one dimension. 20% of the data was used to train the model.

```
from keras.layers import Dense, Dropout, Flatten, Conv1D, Input, MaxPooling1D
from keras.models import Model
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras import backend as K
K.clear_session()

inputs = Input(shape=(8000,1))

conv = Conv1D(8, 13, padding = 'valid', activation = 'relu', strides = 1)(inputs)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)

conv = Conv1D(16, 11, padding='valid', activation = 'relu', strides = 1)(conv)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)

conv = Conv1D(32, 9, padding = 'valid', activation = 'relu', strides = 1)(conv)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)

conv = Conv1D(64, 7, padding = 'valid', activation = 'relu', strides = 1)(conv)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)

conv = Flatten()(conv)

conv = Dense(256, activation = 'relu')(conv)
conv = Dropout(0.3)(conv)

conv = Dense(128, activation = 'relu')(conv)
conv = Dropout(0.3)(conv)

outputs = Dense(len(labels), activation = 'softmax')(conv)

model = Model(inputs, outputs)
model.summary
```



In addition to testing the results on the remaining data, a script was written to allow for the user to input new one-second audio files to test the accuracy of the model.

Results

After training the model through four passes of the convolutional neural network, it is able to take a given audio file and print out what word that audio file says. This is demonstrated by selecting a file at random, printing its label, then printing what the model predicted the audio says.

```
In [16]: from keras.models import load_model
         model = load_model('best_model.hdf5')

In [17]: def predict(audio):
             prob = model.predict(audio.reshape(1,8000,1))
             index = np.argmax(prob[0])
             return classes[index]

In [18]: import random
         index = random.randint(0,len(X_test)-1)
         samples = X_test[index].ravel()
         print("Audio: ",classes[np.argmax(y_test[index])])
         ipd.Audio(samples, rate=8000)
         print("Text: ",predict(samples))

         Audio:  yes
         Text:  yes
```

In addition to the given audio files, through a written script, the user can also record their own audio file to test if the model can accurately predict what they say.

```
In [50]: import sounddevice as sd
         import soundfile as sf

         samplerate = 16000
         duration = 1
         filename = 'go.wav'
         print("start")
         mydata = sd.rec(int(samplerate * duration), samplerate=samplerate, channels=1, blocking=True)
         print("end")
         sd.wait()
         sf.write(filename, mydata, samplerate)

         start
         end

In [53]: samples, sample_rate = librosa.load('stop.wav', sr=16000)
         samples = librosa.resample(samples, sample_rate, 8000)
         ipd.Audio(samples, rate=8000)

Out[53]: ▶ 0:00 / 0:01

In [54]: predict(samples)

Out[54]: 'stop'
```

Discussion

As it can be seen from the results, after sampling the audio files and training the model via a convolutional neural network, the program is now capable of predicting words from the dataset as well as input from the user. The program looks for patterns in the digital audio signals and can tell what the audio says. Given the dataset and simplicity of the program, only ten words can be predicted. However, with a different dataset and a more robust model, it could be extended to allow more words and, potentially, full phrases. In addition to a different dataset, a different neural network could be tested to see if similar results could be achieved. For example, recursive neural networks are also popular for speech recognition. Though this model is simple, it clearly works and gets the proper results. It is also the basis on which most robust speech-to-text models use.

References

https://heartbeat.fritz.ai/a-2019-guide-for-automatic-speech-recognition-f1e1129a141c
https://www.analyticsvidhya.com/blog/2019/07/learn-build-first-speech-to-text-model-python/
https://www.kaggle.com/c/tensorflow-speech-recognition-challenge