Voice Recognition

Seroussi Yishay [✡] Bismuth Samuel [❄] Shaag Yehonatan [✦]

Deep Learning

Abstract

Given a voice recording, by using deep learning, we are able to determine (With high percentages of certainty) from what country the accent of the speaker coming, and then, make assumption about the origin of the speaker.

To achieve such a project, we need a strong data-set built from the website provided by our lecturer. Our data-set includes each one of the next languages : French, Hebrew, USSR and English (UK and USA), five different recording.

Each recording is the next text : " Please call Stella. Ask her to bring these things with her from the store: Six spoons of fresh snow peas, five thick slabs of blue cheese, and maybe a snack for her brother Bob. We also need a small plastic snake and a big toy frog for the kids. She can scoop these things into three red bags, and we will go meet her Wednesday at the train station. " read by a subject

[✡] Student of Computer Science (third year), Ariel University, Ariel 40700, Israel. Id : 305027948. Email: seroussi1@gmail.com

[❄] Student of Computer Science (third year), Ariel University, Ariel 40700, Israel. Id : 342533064. Email: samuelbismuth101@gmail.com

[✦] Student of Computer Science (third year), Ariel University, Ariel 40700, Israel. Id : 308357953. Email: yoshago@gmail.com

1 Data-set

The data set of the website is as the next table:

Adding to each recording, for each subject we got information, such as the next table 1.

birth place	st. laurent d'onay, france (map)
native language	french (fra)
other language(s)	spanish
age, sex	20, female
age of english onset	12
english learning method	academic
english residence	usa
length of english residence	0.4 years

Table 1: Biographical Data

Adding of course the audio.

We decide to take only the native language of the fifth regions (France, USSR, UK, USA and Israel) and the audio that we convert into a WAV file. The rest of the data will be used in order to test our data-set.

An important note is about the audio: Of course the computer don't understand audio such as human does, so, we have to translate the audio into numbers understood by a computer. To do this, and by Lecturer's advice, we use Mel-frequency (mfcc) to handle the audio. To do this, we downloaded all the audio in a WAV files, and we use the module "python speech features" to transform the audio file into a matrix of numbers understanding by the computer.

Then, our actual data set is composed of a CSV table (ready to be read by a python script and easily alterable) with two columns, the first is the region and the second is the link to the WAV file.

2 Work description

To recognize accent, using Tensorflow, we implement a multinomial logistic regression (softmax). First, we recuperate the data from the csv, and convert it into numerous python object, such that each object includes data about the accent of the recorder, and three seconds of audio. The data is splitted into four arrays. Two arrays for the train: one with the data set and the second with the labels and same for the test. 70% of the data is used for the train. In the implementation of the softmax, a lot of parameters can be modified. Our work for now will be to find the best values for these parameters that will optimize the loss function and the test accuracy.

We made some tests trying to improve the results, such as the next table 2.

Test number	Data	Parameter	Remark
1	English,(UK and USA). USSR, French: 40 minutes of recording from 20 different people. Hebrew: 18 minutes of recording from 9 different recorders.	Gradient: 0.00001, features: 1, accuracy: 20%.	At this stage, the goal of the softmax wasn't understood yet. Note that we worked with a matrix of number converted by mfcc.
2	English,(UK and USA). USSR, French: 40 minutes of recording from 20 different people. Hebrew: 18 minutes of recording from 9 different recorders.	Gradient: 0.01, features = 3800, batch: 200, epochs: 5001, accuracy of the train: 100%, accuracy of the test: 35%.	We converted the matrix into an array to make the work easier. Also, we decided to divide each record into numerous records between 2 and 5 seconds.
3	French: 1 hour of recording from 30 people. Hebrew: 18 minutes of recording from 9 different recorders. Russian: 80 minutes of recording from 40 different recorders. UK: 55 minutes of recording from 36 different recorders, USA: 1 hour of recording from 33 people.	Gradient: 0.05, features = 12950, batch: 100, epochs: 500, accuracy of the train: 100%, accuracy of the test: 40%	We divide English into two classes: USA and UK. The conclusion of the result was an over-fitting, so we wanted to add regularization, sadly, without success. Same for normalization.
4	French: 2 hours of recording from 50 people. Hebrew: 38 minutes of recording from 14 different recorders. Russian: 2 hours 20 minutes of recording from 40 different recorders. UK: 90 minutes of recording from 29 different recorders, USA: 2 hour of recording from 50 people.	Gradient: 0.05, features = 6450, batch: 100, epochs: 500, accuracy of the train: 80%, accuracy of the test: 50%	We added more data from some other platform - YouTube. The new data is not necessary the "please call Stella" record. By adding the data we saw an improvement, we need to deal with the over-fitting, to be more exact.

Table 2: Experiences

Our final test accuracy is 50%. Since the guessing is about 20%, we gain 30% by using the softmax. In words, the computer is guessing right once on two. For the future work, and to improve the guessing, it should be interesting to add normalization or/and regularization, adding also more data...