Bachelor's Thesis Project

Even'2017

Spoken Language Identification using Deep Neural Networks

Under Guidance of :
Dr. Shampa Chakraverty

210CO13 Aditya Jain
233CO13 Anmol Pandey
234CO13 Anmol Varshney

Introduction

There has been much literature related to speech recognition in the machine learning community. The most common problem in this area is to translate spoken words into text. However, this approach is generally restricted to the case where the speaker's language is already known. In many situations, such as automated telephone systems, it would be useful for an algorithm to first recognize the speaker's language so that it can translate the speech from the appropriate language. As there has not been much research on this problem, we examine deep neural network based approach to language classification

Data Processing

We obtained our training and testing data from VoxForge , a website providing a collection of audio recordings of speech in various languages.

We wrote code to scrape a large number of 16KHz 16-bit recordings in six languages: English, French, German, Dutch, Italian, and French. Most of the clips are between four and seven seconds in length.

We strip the unvoiced parts of speech because data sanity and computation reduction is a major concern for us We then divide the remaining speech into 30 ms frames, each overlapping by 15ms

We obtain features vector for each frame of the speech, we use the following features for the same:

Mel-Frequency Cepstral Coefficients

Delta Cepstral coefficients

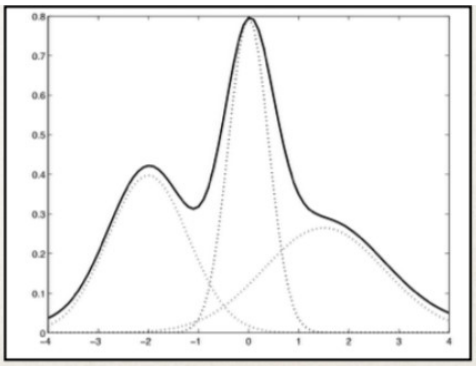
Delta Delta Cepstral Coefficients

Gaussian Mean and Variance (GMM Model)

Gaussian Mixture Model

- GMMs are used to represent frame-based speech features
- Used for estimating acoustic likelihoods
- GMMs are a weighted sum of multivariate Gaussians

$$f(x|\mu,\Sigma) = \sum_{k=1}^{M} c_k \frac{1}{\sqrt{2\pi|\Sigma_k|}} exp[(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)]$$



The basic Aim of Using Gaussian Mixture Model is to minimize the log likelihood function

- Consider log likelihood

$$\ln p(X \mid \mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln p(x_n) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n \mid \mu_k, \Sigma_k) \right\}$$

ML does not work here as there is no closed form solution

Parameters can be calculated using Expectation Maximization (EM) technique

For which it uses the Expectation Minimisation (EM) Algorithm

- From Bayes rule

$$\gamma_k(x) = p(k \mid x) = \frac{p(k)p(x \mid k)}{p(x)}$$

Latent Variable

$$= \frac{\pi_k \mathcal{N}(x \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x \mid \mu_j, \Sigma_j)} \quad \text{where,} \quad \pi_k = \frac{N_k}{N}$$

2. E step. Evaluate the responsibilities using the current parameter values

$$\gamma_j(\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x} \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x} \mid \mu_j, \Sigma_j)}$$

3. M step. Re-estimate the parameters using the current responsibilities

$$\mu_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)} \qquad \Sigma_j = \frac{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)(\mathbf{x}_n - \mu_j)(\mathbf{x}_n - \mu_j)^T}{\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)} \qquad \pi_j = \frac{1}{N}\sum_{n=1}^{N} \gamma_j(\mathbf{x}_n)$$

4. Evaluate log likelihood

$$\ln p(\mathbf{X} \mid \mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k \mathrm{N}(\mathbf{x}_n \mid \mu_k, \Sigma_k)\right\}$$

If there is no convergence, return to step 2.

MFCC Features

- The mel-frequency cepstrum is a representation of an audio signal on the mel scale, a nonlinear mapping of frequencies that down-samples higher frequencies to imitate the human ear's ability to process sound.

In our implementations, we used the first 13 cepstral coefficients as our primary features, as is common in similar applications

Delta Features

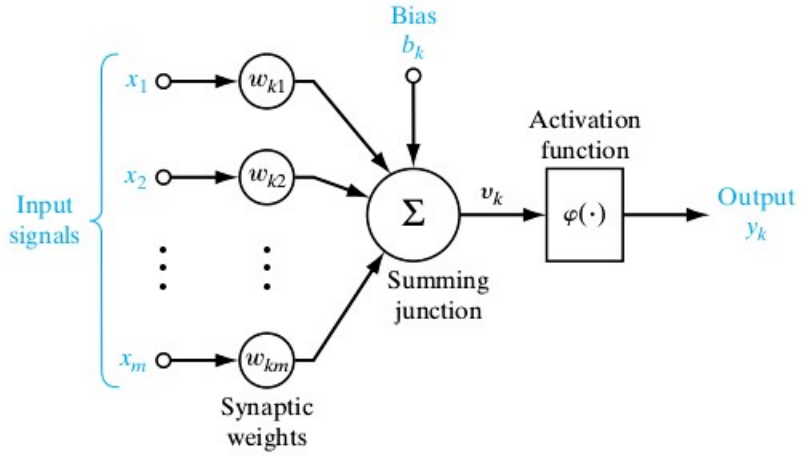
- We also investigated using delta features. Delta features are the first and second time derivatives of the cepstral coefficients, capturing the change of the cepstral features over time, which we hypothesize will be useful in classifying language, since pace is an important factor in language recognition by humans.

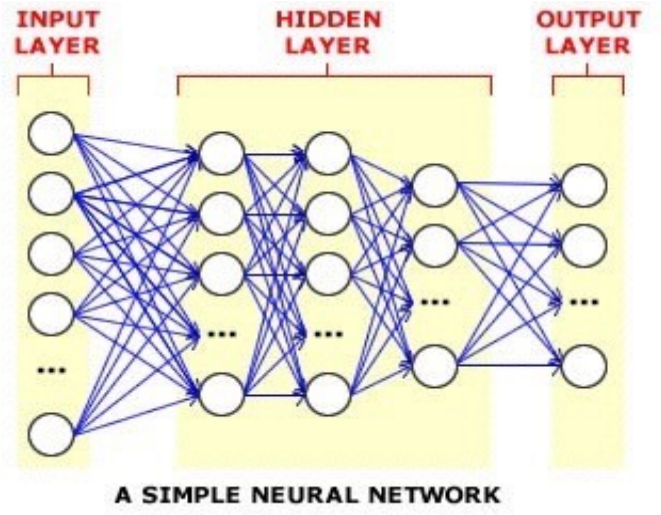
- We can calculate these features as the central finite difference approximation of these derivatives

$$\Delta_{t,i} = \frac{c_{t+1,i} - c_{t-1,i}}{2} \qquad \text{and} \qquad \Delta^2_{t,i} = \frac{c_{t+1,i} - 2c_{t,i} + c_{t-1,i}}{4},$$

Deep Learning

Deep neural network is a feed-forward artificial neural network with multiple hidden units between input and output





A SIMPLE NEURAL NETWORK

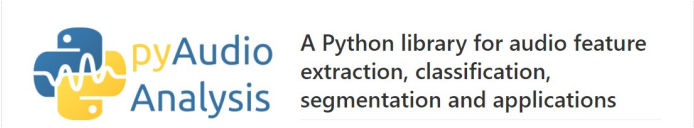
Why Deep Learning

- DNNs have ability to learn complicated feature representations and classifiers jointly
- Learn much better models of data that lie on or near a non-linear manifold
- Performance does not saturate with increase in training data
- Deep learning proven to be better than other Models

DNN Proposed Structure

- Our proposed DNN at the 1st stage will contain 1 hidden layer with 13 MFCC features, 13 Delta MFCC and 13 Delta Delta MFCC.
- The number of outputs is equal to the number of languages to be identified and the outputs are the respective probabilities of the languages.
- We intend to add Formants as our input features to learn from speech pronunciations
- The DNN at the second layers will be trained on features selected for binary classification b/w each pair of languages using LDA and they will be stored in a confusion matrix
- These binary classifiers will have 2 outputs for each language.

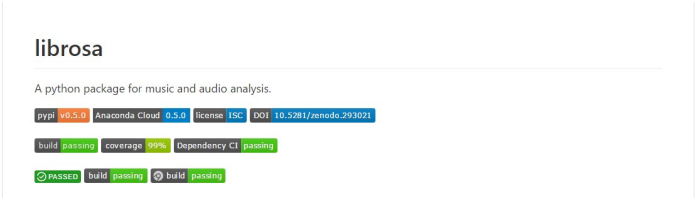
Tools Used



PyAudioAnalysis :

Used for framing and audio extraction.The python library is used to obtain both short term and averaged mid term feautures

We are using it to extract the **MFCC** Features.It takes the audio as wav file and depending on the window size and hop length returns the MFCC coefficients of each feature window

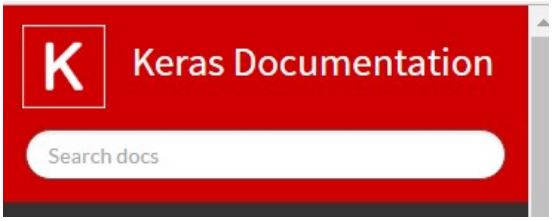


Librosa:

Used for extracting the delta and delta-delta cepstal coefficients

Scikit-learn:

Used for extracting the Gaussian Mixture Models



Keras:

Keras uses Theano as backend for implementing deep neural networks with customizable activation functions and hidden layers

Further Work

Linear Discriminant Analysis(LDA)

It is a composite, 2-stage technique, first reduce dimensionality, then classify.

The objective of LDA is to perform dimensionality reduction while preserving as much of the class discriminatory information as possible

Why to use LDA:

-Shorter training time

-Avoids the curse of dimensionality

-Enhances generalization by reducing overfitting

The generalization of the within-class scatter matrix is

$$S_W = \sum_{i=1}^{C} S_i$$

$$\text{where } S_i = \sum_{x \in \omega_i} (x - \mu_i)(x - \mu_i)^T \text{ and } \mu_i = \frac{1}{N_i} \sum_{x \in \omega_i} x$$

The generalization for the between-class scatter matrix is

$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$$\text{where } \mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{x \in \omega_i} N_i \mu_i$$

For the (C-1) class problem we will seek (C-1) projection vectors wi, which can be arranged by columns into a projection matrix W=[w1|w2|…|wC-1] so that

$$y_i = w_i^T x \Rightarrow y = W^T x$$

where optimal projection matrix W* is the one whose columns are the eigenvectors corresponding to the largest eigenvalues of the following generalized eigenvalue problem

$$W^* = [w_1^* \mid w_2^* \mid \cdots \mid w_{C-1}^*] = \text{argmax}\left\{\frac{W^T S_B W}{W^T S_W W}\right\} \Rightarrow (S_B - \lambda_i S_W) w_i^* = 0$$

The new vector y can be used as our set of reduced feature.