Project: SIV895

# GA and PSO based feature selection for Speaker Recognition

Soumen Basu

---

**Objective and Motivation:**

Speaker recognition is the process of automatically recognizing who is speaking by using the speaker-specific information included in speech waves to verify identities being claimed by people accessing systems. By checking the voice characteristics of utterance, this system makes it possible to authenticate their identity and control access to services. This will add extra level of security to any system.

Speaker recognition or voiceprint recognition is an application based on physiological and behavioral characteristics of the speaker's voice. Speaker recognition takes care of the unique features of speech are analyzed to identify the speaker, these unique features will be extracted and converted to digital symbols, and then these symbols are stored as that person's character template. This template is stored in a computer database. User authentication is processed inside the recognition system to identify matching or not.

The number of features presented in speech signals is often large. It is thus required to minimize the set of features which will be used for recognition task. In this report, we present a comprehensive study of GA and PSO based feature selection algorithms for speaker recognition.

**Background study:**

System Overview :- For getting the speaker identity , there are two stages.

1. Feature extraction
2. Classification and Feature matching

Feature extraction is a special form of dimensionality reduction, and here in our project we need to do dimensionality reduction for the input speech ; we will do that by extracting a specific features from the speech, these features carry the characteristics of the speech which are different from one speaker to another, so these features will play the major role in our project , as our mission is to identify the speaker and make a decision that highly depends on how much we were successful in extracting a useful information from the speech in a way enables our system to differentiate between speakers and identify them according to their features .

there are some question which we are interested in while recognition process.

What do you want to extract from the input speech ?

- Feature vectors .

What these feature vectors represent ?

- Formants which carry the identity of the speech.

What are the formants ?

- To identify these formants here a brief discussion to the speech production in the human body is given:

The three main cavities of the speech production system are nasal, oral, and pharyngeal forming the main acoustic filter, The form and shape of the vocal and nasal tracts change continuously with time, creating an acoustic filter with time-varying frequency response. As air from the lungs travels through the tracts, the frequency spectrum is shaped by the frequency selectivity of these tracts. The resonance frequencies of the vocal tract tube are called formant frequencies or simply formants. which depend on the shape and dimensions of the vocal tract. The speed by which the cords open and close is unique for each individual and define the feature and personality of the particular voice.

## Feature Extraction Algorithms:

Several feature extraction algorithms are used to this task such as linear predictive coefficients (LPC), linear predictive cpestral coefficients (LPCC), mel frequency cpestral coefficients (MFCC) , and human factor cpestral coefficient (HFCC).
We used the (mfcc) algorithm to extract the features, we choose it for the following reasons :
1.one of the most important features, which is required among various kinds of speech applications.
2.shows high accuracy results for clean speech .
3 . they can be regarded as the "standard" features in speaker as well as speech recognition. However, experiments show that the parameterization of the MFC coefficients which is best for discriminating speakers is different from the one usually used for speech recognition applications.
4. The most common algorithm that used for speaker recognition system.

**Inside the mfcc algorithm:**
1)Preprocessing :-
To enhance the accuracy and efficiency of the extraction processes, speech signals are normally pre-processed before features are extracted. Speech signal pre-processing covers digital filtering and speech signal detection. Filtering includes pre-emphasis filter and filtering out any surrounding noise using several algorithms of digital filtering.

***Pre-emphasis filter:***
In general, the digitized speech waveform has a high dynamic range and suffers from additive noise. In order to reduce this range, pre-emphasis is applied. This pre-emphasis is done by using a first-order FIR high-pass filter.
In the time domain, with input $x[n]$ and $0.9 \le a \le 1.0$, the filter equation
$y[n] = x[n]-a \, . \, x[n-1]$.
And the transfer function of the FIR filter in z-domain is:
$H ( Z ) = 1 - \alpha \, . \, z - 1 \, , 0.9 \le \alpha \le 1.0$
Where $\alpha$ is the pre-emphasis parameter.
The pre-emphasizer is implemented as a fixed coefficient filter or as an adaptive one, where the coefficient a is adjusted with time according to the auto-correlation values of the speech.
The aim of this stage is to boost the amount of energy in the high frequencies. The drop in energy across frequencies (which is called spectral tilt) is caused by the nature of the glottal pulse. Boosting the high frequency energy makes information from these higher formants available to the acoustic model. The pre-emphasis filter is applied on the input signal before windowing.
***Framing:***
First we split the signal up into several frames such that we are analyzing each frame in the short time instead of analyzing the entire signal at once, at the range (10-30) ms the speech signal is for the most part stationary .
Also an overlapping is applied to frames. Here we will have something called the Hop Size. In most cases half of the frame size is used for the hop size. The reason for this is because on each individual frame, we will also be applying a hamming window which will get rid of some of the information at the beginning and end of each frame. Overlapping will then reincorporate this information back into our extracted features.
***Windowing:***
It is necessary to work with short term or frames of the signal. This is to select a portion of the signal that can reasonably be assumed stationary. Windowing is performed to avoid unnatural discontinuities in the speech segment and distortion in the underlying spectrum . The choice of the window is a tradeoff between several factors. In speaker recognition, the most commonly used window shape is the hamming window.
The multiplication of the speech wave by the window function has two effects :-

1-It gradually attenuates the amplitude at both ends of extraction interval to prevent an abrupt change at the endpoints .
2-It produces the convolution for the Fourier transform of the window function and the speech spectrum .
Actually there are many types of windows such as : Rectangular window , Hamming window , Hann window , Cosine window , Lanczos window , Bartlett window (zero valued end-points) , Triangular window (non-zero end-points) , Gauss windows .
we used hamming window the most common one that being used in speaker recognition system. The use for hamming windows is due to the fact that mfcc will be used which involves the frequency domain(hamming windows will decrease the possibility of high frequency components in each frame due to such abrupt slicing of the signal).

***Fast Fourier Transform:***

To convert the signal from time domain to frequency domain preparing to the next stage ( mel frequency wrapping ).

The Mel Scale

The speech signal consists of tones with different frequencies. For each tone with an actual Frequency, f, measured in Hz, a subjective pitch is measured on the 'Mel' scale. The mel-frequency scale is a linear frequency spacing below 1000Hz and a logarithmic spacing above 1000Hz.
we can use the following formula to compute the mels for a given frequency f in Hz:
mel(f)= 2595*log10(1+f/700).
One approach to simulating the subjective spectrum is to use a filter bank, one filter for each desired Mel frequency component. The filter bank has a triangular band pass frequency response, and the spacing as well as the bandwidth is determined by a constant mel-frequency interval.

Mel frequency analysis

- Mel-Frequency analysis of speech is based on human perception experiments.
- Human ears, for frequencies lower than 1 kHz, hears tones with a linear scale instead of logarithmic scale for the frequencies higher than 1 kHz.
- The information carried by low frequency components of the speech signal is more important compared to the high frequency components. In order to place more emphasize on the low frequency components, mel scaling is performed.
- Mel filterbanks are non-uniformly spaced on the frequency axis, so we have more filters in the low frequency regions and less number of filters in high frequency regions .

***Cepstrum***

In the final step, the log mel spectrum has to be converted back to time. The result is called the mel frequency cepstrum coefficients (MFCCs). The cepstral representation of the speech spectrum provides a good representation of the local spectral properties of the signal for the given frame analysis. Because the mel spectrum coefficients are real numbers(and so are their logarithms), they may be converted to the time domain using the Discrete Cosine Transform (DCT).
Since the speech signal represented as a convolution between slowly varying vocal tract impulse response (filter) and quickly varying glottal pulse (source), so, the speech spectrum consists of the spectral envelop(low frequency) and the spectral details(high frequency).
Now, our goal is to separate the spectral envelope and spectral details from the spectrum.
It is known that the logarithm has the effect of changing multiplication into addition. Therefore we can simply converts the multiplication of the magnitude of the fourier transform into addition.
Then, by taking the inverse FFT or DCT of the logarithm of the magnitude spectrum, the glottal pulse and the impulse response can be separated.
We will use DCT , why DCT ?
The signal is real ( we took the magnitude) with mirror symmetry.
The IFFT needs complex arithmetic, the DCT does not.
The DCT implements the same function as the FFT more efficiently by taking advantage of the redundancy in a real signal.

The DCT is more efficient computationally.
The MFCCs may be calculated using this equation:

$$\tilde{C}_n = \sum_{k=1}^{K} (\log \tilde{S}_k) \left[ n\left(k - \frac{1}{2}\right)\frac{\pi}{K} \right],$$

where n=1,2,....K

The number of mel cepstrum coefficients, K, is typically chosen as (10-15). The first component, *c*~0 , is excluded from the DCT since it represents the mean value of the input signal which carries little speaker specific information. Since the log power spectrum is real and symmetric, inverse FFT reduces to a Discrete Cosine Transform (DCT). By applying the procedure described above, for each speech frame of about 30 ms with overlap, a set of mel-frequency cepstrum coefficients is computed. This set of coefficients is called an *acoustic vecto*r.
These acoustic vectors can be used to represent and recognize the voice characteristic of the speaker. Therefore each input utterance is transformed into a sequence of acoustic vectors. The next section describes how these acoustic vectors can be used to represent and recognize the voice characteristic of a speaker.

**Nature of Dataset:**

We have worked with a dataset of 8100 samples with 543 features. The data samples contain 5 classes for classification.

**GA based feature selection algorithm:**

```
Procedure GA
{
    t = 0;
    Generate a chromosome as a random binary string of size
    equal to number of features.
    Initialize P(t) = Set of chromosomes;
    Evaluate_Accuracy P(t); // This is the fitness function
    While (Not Done)
    {
           Parents(t) = Select_Parents(P(t));
           Offspring(t) = Cross_and_Mutate(Parents(t));
           Evaluate_Accuracy(Offspring(t));
           P(t+1)= Select_Survivors(P(t),Offspring(t));
           t = t + 1;
      }
}
```

Fig: Outline of the basic GA-based algorithm

Brief descriptions of the functions are given below,

- `Select_Parents()`: This function finds the set of parent chromosomes from which the childs would be derived. I used the ***Linear Rank Selection*** (LRS) method to select the parents. This is a simple method that tries to overcome the premature convergence and local minima problems.

- `Cross_and_Mutate()`: This function finds the set of offprings by ***single-point crossover*** and mutation. The cross-over point is generated at random. The number of bits to mutate is generally controlled by a ***mutation rate*** parameter.

- `Evaluate_Accuracy()`: Using this function we calculate the fitness value for each chromosome. The fitness value is the classification accuracy by using the chromosome as feature vector mask.

- `Select_Survivors()`: This function finds the set of survivor chromosomes for next iteration. The method I use is a variant of ***steady-state replacement.*** Few worst valued parent is replaced by the best-valued offsprings.

*Note:* The ***Linear Rank Selection (LRS)*** is a variant of Rullet While Selection that tries to overcome the drawback of premature convergence of the GA to a local optimum. It is based on the rank of individuals rather than on their fitness. The rank n is accorded to the best individual whilst the worst individual gets the rank 1. Thus, based on its rank, each individual ***i*** has the probability of being selected given by the expression,

$$p(i) = rank(i) / (n*(n-1))$$

Once all individuals of the current population are ranked, the LRS procedure can be implemented according to the following pseudocode

```
LRS_ pseudo-code
 {
  • Calculate the sum $v = \frac{1}{n-2.001}$;
  • For each individual $1 \le i \le n$ do {
      - Generate a random number $\alpha \in [0, v]$;
      - For each $1 \le j \le n$ do {
          - If ( $p(j) \le \alpha$ ) {
              - Select the $j^{th}$ individual;
              - Break;
            }
        }
    }

  }
```

**PSO based feature selection algorithm:**

```
Procedure PSO {
     t = 0;
     Generate a particle as a random binary string of size equal to number of
     features.
     Initialize P(t) = Set of particles;
     // This is the initial fitness values
     Calculate_Fitness(P(t));
     While (t<max_iteration) {
          For each particle:
                         Calculate_Fitness();
                         Update_pbest_and_gbest();
                         Update_velocity();
                         Update_Position();
          t = t + 1;
     }
}
```

Fig: Outline of the basic PSO-based algorithm

Following are the breif descriptions of the functions,

- `Calculate_Fitness()`: This function calculates the fitness value for each particle, that is the accuracy obtained using the feature mask defined by that particle.

- `Update_pbest_and_gbest()`: This function simply updates the local_best fitness values for each of the particle. The best fitness value so far is recognized as the global best value.

- `Update_velocity()`: Velocity is updated using the following equation, for each particle, the velocity in phase i+1 is given by,

```
v[i+1] = c0 *v[i] + c1 * rand() * (pbest[i] - pos[i]) + c2*rand()*(gbest - pos[i])
```

I have chosen

- `Update_Position()`: pos[i] = pos[i-1] + v[i-1]

The velocity value is continuous, so is the position value. We translate this back to binary by the use of ***sigmoid*** function.

**Results obtained for GA-based selection:**

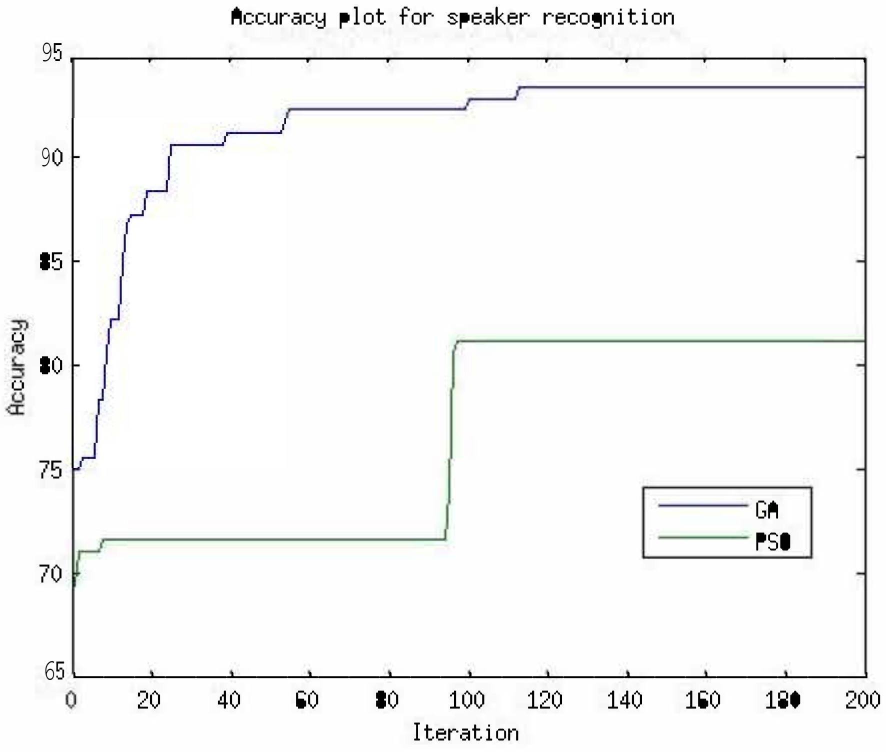
The results that I obtained using GA, is discussed below.
***max_iteration = 100***
***population size = 10***

1. *Accuracy:* The accuracy of the best chromosome increases in a step function like pattern for both datasets. The final accuracy is 92.4%.
2. *Number of Features Selected*: The number of features selected is 33
3. *Convergence*: Convergence is quite fast. After 120 iterations.

**Results obtained for PSO-based selection:**

The results that I obtained using PSO, is discussed below.
***max_iteration = 100***
***number of particles = 10***
***c0 = 0.12***
***c1 = 1.9***
***c2 = 2.1***

1. *Accuracy:* The accuracy of the best chromosome increases in a step function like pattern for both datasets. The final accuracy is 81.68%.
2. *Number of Features Selected:* The number of features selected is 27
3. *Convergence:* Convergence is quite fast. After 100 iterations.


Accuracy plot for speaker recognition
Accuracy
Iteration
95
90
85
80
75
70
65
0
20
40
60
80
100
120
140
160
180
200
GA
PSO