speaker recognition

TP Biometrics

Guangyue CHEN

ENSIIE

02/05/2019

Introduction

This TP is meant to show a example of speaker recognition with the GMM algorithm. It use the database **Validb**, which contains 1060 audio files from 106 speakers, in 5 environment conditions and with 2 text spoken.

Parameterization

The first step is extracting the acoustic parameters from the sound signals. With the help of function <Visualization Parameters>, we can see the original signal spectrum, the energy and the parameter spectrum. Here we can see that the spectrum of energy varies according to changes in signal amplitude. The higher signal amplitude will have higher energy.

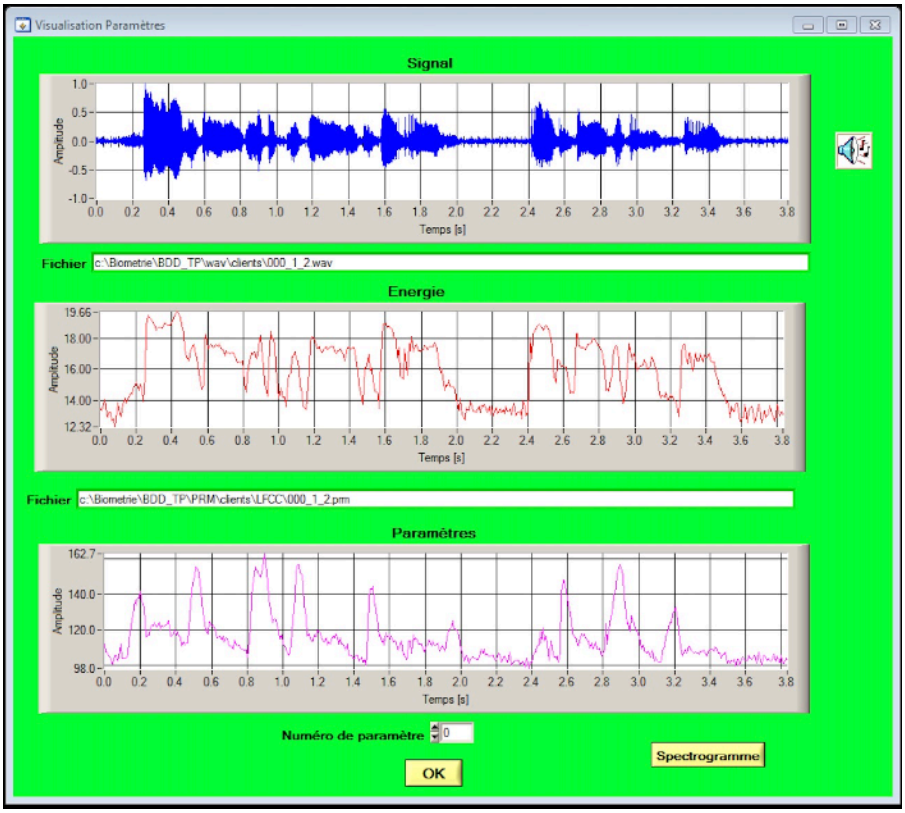

Figure2

And the spectrum of parameters show us for different frequency, how the signal looks like. Then by using the function <Spectrogram>, we can get the variation of the signal spectrum over time.

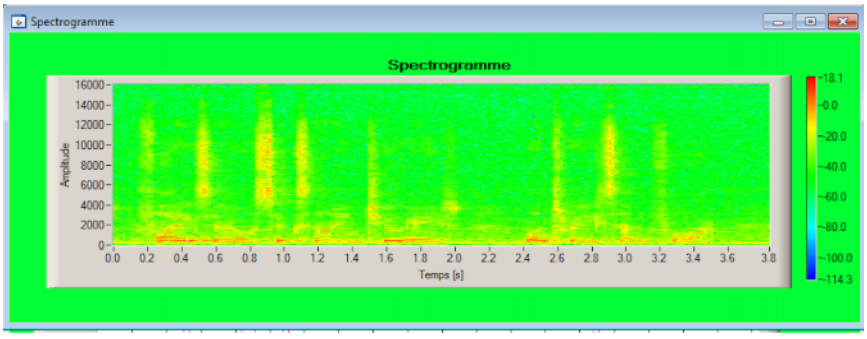

Figure2: a example Spectrogram

The Spectrogram combine the spectrums of 16 parameters(which is from 0 to 15). Here the red point means high amplitude at that time(horizontal axis) and at that frequency(Vertical axis) .

After analyse more files, I compare the spectrums of one sentence in different conditions, although there are some noises(the signals and the energies are so different), the spectrograms are similar. It won't change much.(Figure 3 and 4)

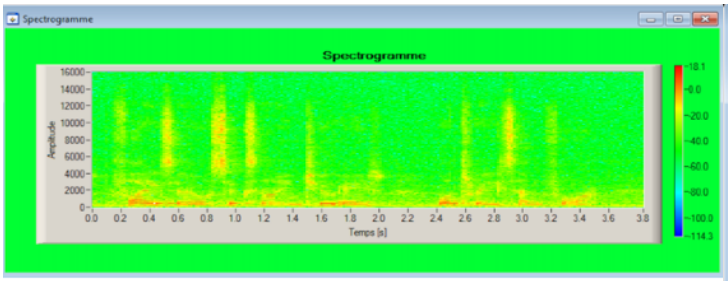

Figure3: 1st text of a woman

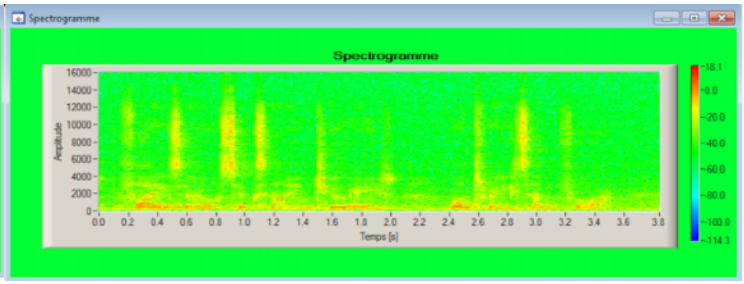

Figure4: 1st text of a woman with 3rd noise

Compare the spectrums of different text, the spectrograms could show us a big difference.(Figure 5 and 6) But we find the difference isn't enough.

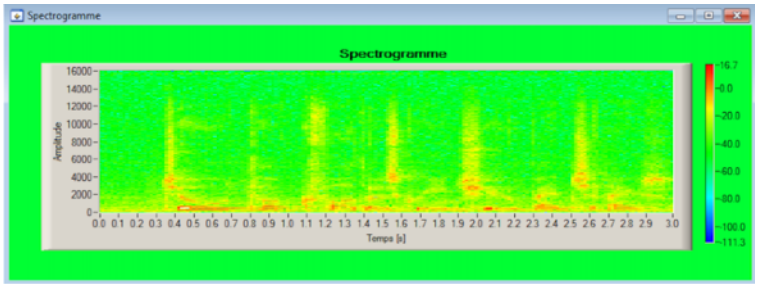

Figure5: 2nd text of 1st woman

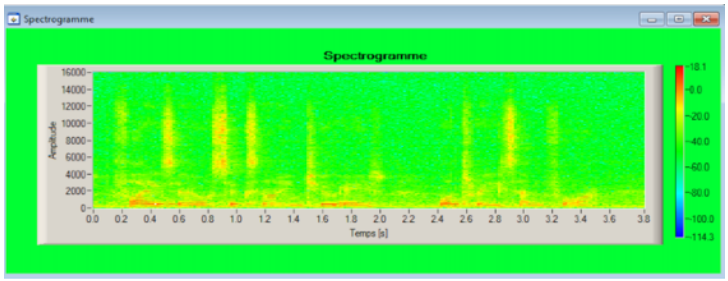

Figure6: 1st text of 1st woman

Energy Detector

This step aims at detecting the periods of silence in the files.

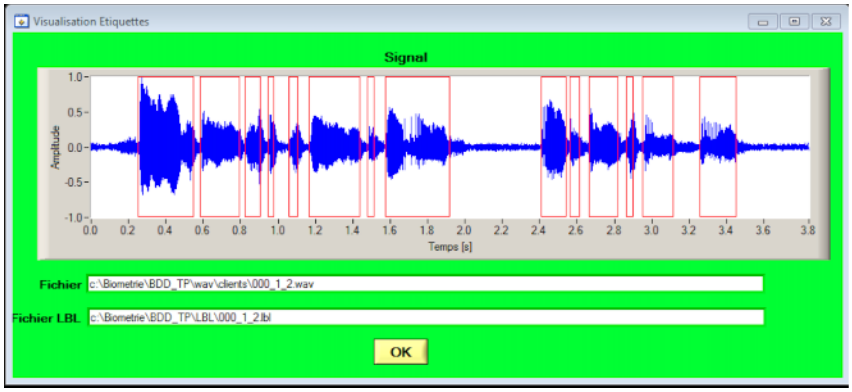

Figure7: Energy Detector without noise

For the signal without noise, it is easy to detect the syllable. But if there are some noises, the result would be so terrible.(Figure 8)

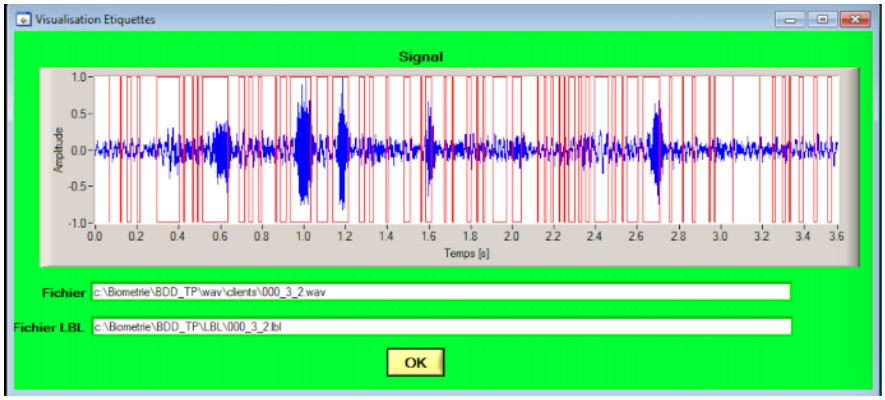

Figure8: Energy Detector with noise

Parameter Normalization

This step performs a normalization of the acoustic parameters. It could give us a better performance of the data.

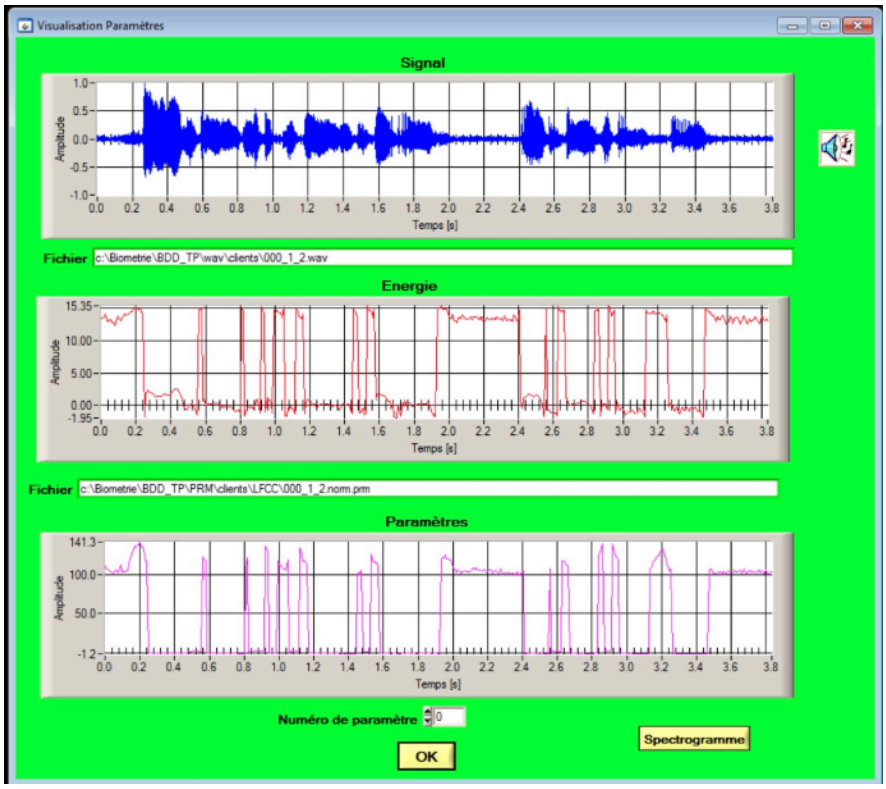

Figure9: Energy Detector with noise

After the normalization, when the signal is in silence, the spectrum of energy is high. And the spectrums of parameter are changed too. Compare with the figure1, the spectrum is more obvious and easier to treat. Compare the

Spectrograms we can find that normalization Expand the difference.(From figure 10 and 11, the points with high amplitude are more than before)

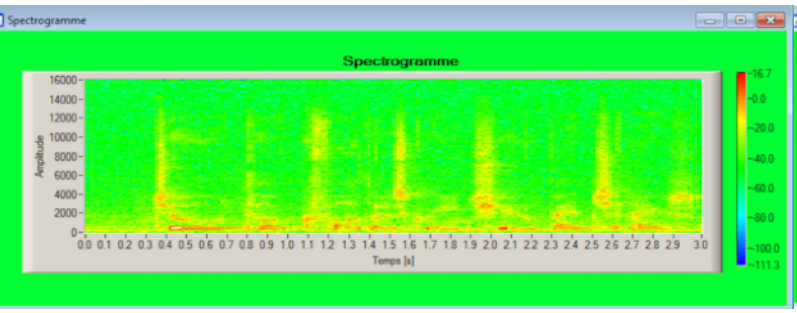

Figure10: 2nd text of 1st woman with normalization

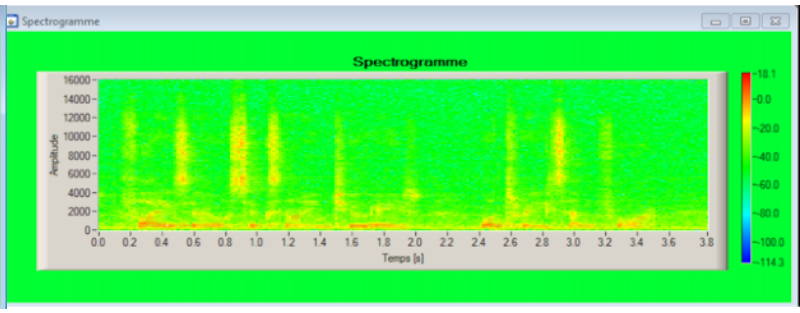

Figure11: 1st text of 1st woman with normalization

Learning the world model

We get a world model here, from the analysis, we find that there are 16 Gauss distribution. And the most representative is the first one whose weight is *0.13291590860141284*. After 15 iterations, it's mean converge to *0.46411288095770276*.

```
<MixtureGD version="1" id="#1" distribCount="16" vectSize="16">
    <DistribGD i="0" weight="0.13291590860141284" cst="7.3894925646845422e-006"
    det="0.0031037938860226937">
        <covInv i="0">2.0000000001461786</covInv>
        <covInv i="1">2.0000000003867071</covInv>
        <covInv i="2">2.000000000056517</covInv>
        <covInv i="3">1.9999999998806723</covInv>
        <covInv i="4">1.876202150595597</covInv>
        <covInv i="5">1.4454032833776334</covInv>
        <covInv i="6">1.6273345066249285</covInv>
        <covInv i="7">1.2712555193733608</covInv>
        <covInv i="8">1.2634446410517624</covInv>
        <covInv i="9">1.138640730209274</covInv>
        <covInv i="10">1.3879476330287035</covInv>
        <covInv i="11">1.3149651669828801</covInv>
        <covInv i="12">1.2672868501491161</covInv>
        <covInv i="13">1.0039538780841912</covInv>
        <covInv i="14">1.1224325309294338</covInv>
        <covInv i="15">0.95725930265082126</covInv>
        <mean i="0">-0.059994271958075086</mean>
        <mean i="1">0.93230659846815633</mean>
        <mean i="2">0.82816765009584148</mean>
        <mean i="3">0.87602435421923663</mean>
        <mean i="4">0.89909935848067879</mean>
        <mean i="5">0.7181695295928946</mean>
        <mean i="6">0.80664364871266825</mean>
        <mean i="7">0.8257141682666892</mean>
        <mean i="8">0.9237801876508015</mean>
        <mean i="9">0.84656970820197697</mean>
        <mean i="10">1.0499512401645767</mean>
        <mean i="11">1.017869183895094</mean>
        <mean i="12">0.94532494022637159</mean>
        <mean i="13">0.80422243634088664</mean>
        <mean i="14">0.62762772925334165</mean>
        <mean i="15">0.46411288095770276</mean>
    </DistribGD>
```

Figure12: the world model

Independent text system

And then we learn client models for each speaker with using the world model. Compare a client model with the world model, we can see that the mean values of Gauss distributions are changed, when the weights of distributions or other values are still the same.

```
<MixtureGD version="1" id="000" distribCount="16" vectSize="16">
    <DistribGD i="0" weight="0.13291590860141284" cst="7.3894925646845422e-006"
    det="0.0031037938860226928">
        <covInv i="0">2.0000000001461786</covInv>
        <covInv i="1">2.0000000003867071</covInv>
        <covInv i="2">2.00000000056517</covInv>
        <covInv i="3">1.9999999998806723</covInv>
        <covInv i="4">1.876202150595597</covInv>
        <covInv i="5">1.4454032833776334</covInv>
        <covInv i="6">1.6273345066249285</covInv>
        <covInv i="7">1.2712555193733608</covInv>
        <covInv i="8">1.2634446410517624</covInv>
        <covInv i="9">1.138640730209274</covInv>
        <covInv i="10">1.387947630287035</covInv>
        <covInv i="11">1.3149651669828801</covInv>
        <covInv i="12">1.2672868501491161</covInv>
        <covInv i="13">1.0039538780841912</covInv>
        <covInv i="14">1.1224325309294338</covInv>
        <covInv i="15">0.95725930265082126</covInv>
        <mean i="0">-0.24566468886564038</mean>
        <mean i="1">0.82286480023135355</mean>
        <mean i="2">0.70011844339628182</mean>
        <mean i="3">0.91021190501307325</mean>
        <mean i="4">1.405229119060772</mean>
        <mean i="5">1.0727880963498726</mean>
        <mean i="6">0.43220612656780966</mean>
        <mean i="7">0.95986702052425488</mean>
        <mean i="8">1.583878325350635</mean>
        <mean i="9">1.1045847325420792</mean>
        <mean i="10">1.0710277283063778</mean>
        <mean i="11">1.0969433980238035</mean>
        <mean i="12">1.3036810632022904</mean>
        <mean i="13">1.1180955950759763</mean>
        <mean i="14">0.67892018344413929</mean>
        <mean i="15">0.677280178226495</mean>
    </DistribGD>
```

Figure13: 000 client model

And then we compute the likelihood, and we get the ROC curve. From the curve, we can have a 10% FAR when the FRR is around 42.5%.

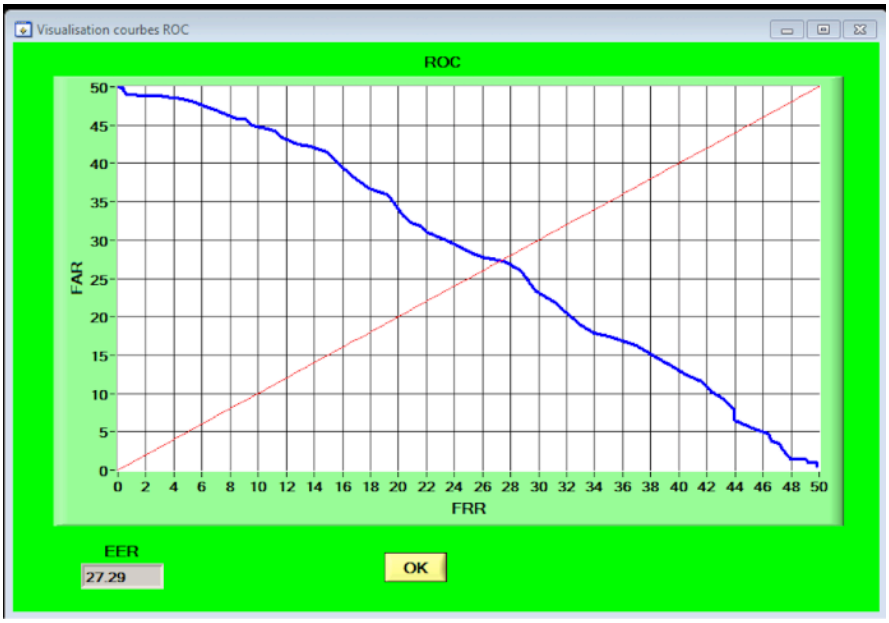

Figure14: ROC independent

And the limits of variations are 0% and 50%. And the intersection point of EER and the ROC curve is around FAR 27% and FRR 27%.

dependent text system

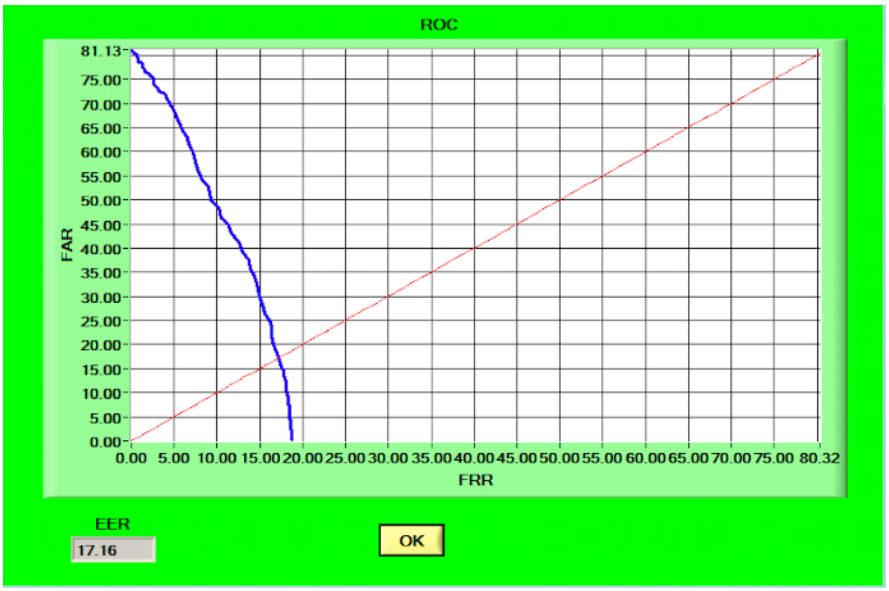

Figure15: ROC dependent

For dependent text system, the limits of FAR is 0% and 81.13%. The limits of FRR is 0% and 18.87%. Compare with the independent system, this curve is steeper. When when get a a 10% FAR, the FRR is around 20%, which is smaller than the FRR of independent system.

And the intersection point of EER and the ROC curve is around FAR 18% and FRR 18%. Which is smaller than the independent text system, so we can say that it has a better performance.