NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY
Department of Computer and Information Science

IT3105 : Artificial Intelligence Programming

Project 2 : Speech Recognition

Alexandre Balon-Perin

Professor: Helge Langseth

Academic Year 2011 - 2012

Contents

Chapter 1

Sound Processing

As we have seen in the lecture, the sound is dynamic and noisy. So, a robust representation of the sound signal must be found.

Firstly, the signal has been framed in pieces of 1/100 seconds. The most important changes in a speech happen at no more than 100Hz. As a consequence, none of the important information is lost when framing the signal.

Afterwards, the fourier transform was applied to each of the frames. The fourier transform takes the data from the time domain to the frequency domain. The two most relevant frequencies were then extracted thanks to the peakdet function (the green dots on the figure 1.1). The red peaks are mostly generated by noise and contain little information about the speech.

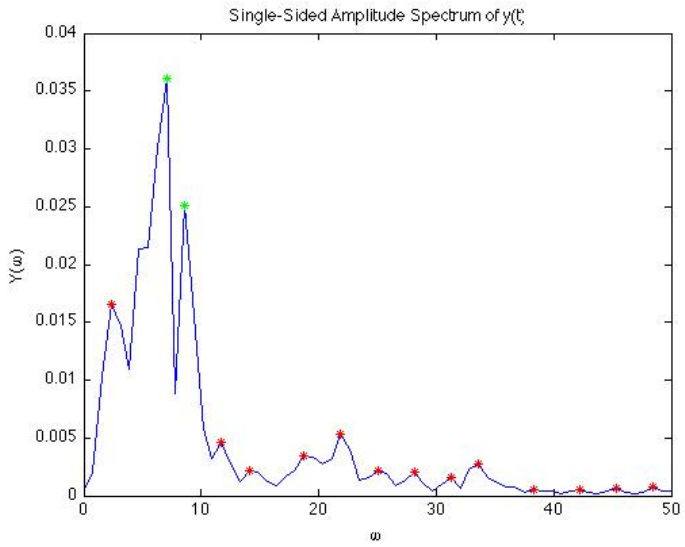

Figure 1.1: Example of a frame in the spectral domain and its peaks

Finally, the log-energy was added. The idea behind the log-energy is to make the distinction between the voiced data, the unvoiced data(the noise) and the silence. The energy of the voiced data is much higher than the energy of the silence. Also the energy of unvoiced data is lower than the one of the voiced data.

Chapter 2

Code Structure

2.1 Part I: Encoding

The class Encoding is in charge of reading and concatenate the files, and extracting the relevant features of the sound signal.

- **Input**: the file path
- **Ouput**: the features extracted
- **Functionalities**:

  First, the file is read

  Second, the signal is framed and a hamming window is applied to the frame

  Finally, the fourier transform is processed and the features are extracted

2.2 Part II: Training

2.2.1 HMM

The HMM class implements the Hidden Markov Model

- **Input**: The word, the number of hidden states and the distribution used
- **Ouput**: The forward and backward messages and the likelihood
- **Attributes:**

  dynModel: is the transition matrix

  priorHidden: is the prior distribution

- **Functionalities**:

  The forward-backward algorithm is implemented in this class

  The transition function and the prior distribution are updated here as well at the M-step of the EM algorithm

2.2.2 Classifier

The Classifier is the main class, its goal is to train the HMM and test the result of the training.

- **Input**: none
- **Ouput**: A trained HMM and the result of the classification
- **Attributes:**

  em: instance of the EM class

  models: a vector of HMM objects

  encoder:instance of the Encoding class

2.2.3 ObsModel

The class ObsModel implements the observation model (the B matrix).

- **Input**: The name of the distribution chosen (value: Gauss 'or' MixtGauss)
- **Ouput**: The likelihood of the observation: $P(O_k|S_k = i)$
- **Functionalities**:

  Implement the multivariate gaussian distribution and the multivariate mixture gaussian. The user has to choose which distribution to use in the class Classifier

  Update the values of μ and σ

2.3 Part III: EM

The EM class implements the EM algorithm.

- **Input**: none
- **Ouput**:none
- **Functionalities**:

  Run the Em algorithm and update $\gamma_t(i)$ and $\xi_t(i, j)$

Chapter 3

Results

3.1 What went wrong

Unfortunately, the result were not as expected. Despite, checking the code for an error of implementation during hours, I could not find one. The code crash after a few iteration because the values of sigma reach NaN. As a consequence, I could not test my system until the end. The values of the forward and backward messages seems correct at the beginning as well as γ and ξ. However, the transition matrix has values higher than 0 which does not make much sense. The formula of this matrix implies the division between the sum over all t values of ξ and the sum over all t values of γ. Therefore, either γ has too small values or ξ has too big values.

3.2 Improvement

A working system could be nice... The addition of the technique log-sum-exp for the product of very small values could maybe solve the problem of the NaN.