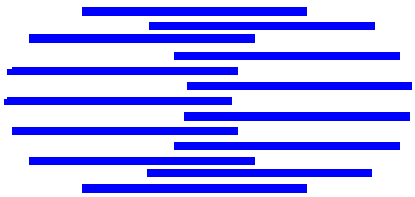
IDIAP COMMUNICATION



Rebuilding Speech Recognition on Windows

Haiyan Wang
IDIAP-Com 01-09

JANUARY 2001

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 – 27 – 721 77 11
fax +41 – 27 – 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

1. INTRODUCTION

In InfoVox project, speech recognition block is designed to process the raw audio data into interpretable sentences. The raw audio data is recorded by telephonic block. The speech recognition block employs the automatic speech recognition technology developed in IDIAP to extract sentences from these raw audio data. The extracted sentences are then output to language processing block.

2. ENVIRONMENT

2.1 Developing Environment

Software - "*Windows NT*", C/C++ language under "*Microsoft Visual Studio*"

Hardware – PC,

2.1.1 Technology Requirements

Automatic Speech Recognition – is based on a large vocabulary continuous speech recognition system, adapted to Swiss-French, and running on Windows NT platform. The recognizer is based on a state-of-the-art hybrid HMM/ANN system, using hidden Markov models (HMM) and artificial neural networks(ANN), to model phonetic units.

Due to the requirements of InfoVox Project, Speech recognition block is consisted of three essential parts: Rasta, Mlp, and Decoder.

RASTA (relative spectral analysis) extracts acoustic vectors from overlapped windows of audio data. Each acoustic vector has 12 coefficients + 1 energy. The width of the window is 30ms of which the center shifts every 10ms.

MLP (Multi Layer Perception, or ANN = Artificial Neural Network) associates with each acoustic vector a relevant phoneme probability.

DECODER, or NOWAY, rebuilds the words and sentences from phonemes according to syntactic and semantic rules respectively.

3. DEVELOPING AND REBUILDING

The developing of speech recognition block is consisted of two parts: transporting the speech recognition block from Unix to Windows NT; optimizing using the features of Windows.

3.1 Porting from Unix to PC

Due to the differences between Unix and Windows and also the differences between compliers, it is not avoidable to change the source code itself. There are mainly five kinds of differences between Unix version source code and windows version code: data formation, function name, calling method, lost functions, old bugs.

3.1.1 Data formation

The binary data is stored in big endian format on Unix while in little endian format on Windows NT platform. Before reusing any binary data from Unix, the bytes'order should be reversed. For example:

```
little endian integer
[ 00 01 02 03]

big endian integer
[ 03 02 01 00]
```

3.1.2 Function name

Some common library functions do not share the same names on Unix and windows, such as popen(), pclose(), rindex() and so on. Their names on windows are individually _popen(), _pclose(), strrchr(). Because there is no such a list that can tell you the windows version name of each function, you must make sure that both names refer to the same function.

3.1.3 Calling method

For some functions, Unix and windows employ different default calling parameters, for example, the function to open a file "fopen". Without an explicit declaration when call the function "fopen", Unix opens a binary file while windows opens a text file. If a file is not opened correctly, some running error will occur unpredictably.

3.1.4 Lost function

Not all the functions on Unix has its copy version on windows. Some functions, like rint() and getopt(), have to be implemented by yourself.

3.1.5 Old bugs

Unix does not check the same kinds of compiling or running errors and warnings as Windows, for example, the multiple definitions. Some bugs in the original code cause running errors though they are not explicitly

appearing on Unix. For example: if a file is opened several times without been closed, it would cause an error on NT.

3.1.6 Porting verification

The executable program of the speech recognition block has been verified with the Unix version program. The same raw audio data is feed to both ported windows version program and the original one on Unix paralleled with the same running parameters. The running parameters are listed in appendix A. Nine raw audio samples are chosen to perform this verification. The verification results are the same as those on Unix, which show that the porting of speech recognition block is successful.

3.2 Optimizing using the features of Windows.

Since the final goal of the InfoVox project is to have a real-time system, it’s important to increase the running speed of speech recognition block.

3.2.1 Exchange data through buffer

The internal results of Rasta, MLP, and Decoder is transfer with files in the Unix version. On windows, the read and write file operations take 100 times more than buffer process. The following section describes how to change model interface to buffer instead of file.

3.2.2 Data form of transport

Firstly, you must know the data form of transport among each model.

Between Rasta and Mlp, transport several groups of 39 float data,([12 coefficients + 1 energy] [12Δ coefficients + 1Δ energy] [12ΔΔ coefficients + 1ΔΔ energy]).

Between Mlp and Decoder, transport several groups of 36 float data, which are values of phonemes probability, and in front of each group, There is a integer which represents order number.

3.2.3 Function position

Secondly, you have to find the functions which do operation of reading or writing file.

3.2.4 Changing method

Thirdly, you should create global transporting buffer, and remove the sentences which read or write file. For sentences of reading file, replace with setting values of input buffer to temperate variables; For sentences of writing file, replace with setting values of temperate variables to output buffer.

There is one thing should be emphasized: In the optimized version of application, has ignored the integer which represents order number for output of Mlp. Thus makes convenient for buffer transporting.

3.2.5 Transporting verification

At last, You must test if the new results of buffer version are the same as those of file version. You can also compute the running time for nine audio samples.

The results and running time of the nine raw audio samples are listed in appendix B.

3.2.6 Using Multi-thread

Windows allows building multiple concurrent execution threads running simultaneously. At the original period of developing, we've implemented a parallel control program integrated with RASTA and MLP using Multi-thread technology. A *mutex* was used to protect a shared resource from simultaneous access by multiple threads. Nevertheless, the running tests showed that the performance of the application in fact decreased. After investigation and analysis, it was concluded that it is impractical to build multi-threading between RASTA and MLP. The running speed of RASTA and MLP are quite different, so it is almost impossible to realize parallel operation. Moreover, Creating Multi-thread requires extra system expense to implement multi-threading.

However, this multi-thread application provides a framework for the *InfoVOX* system, and it may have potential usage in other applications.

APPENDIX A

PARAMETRES OF the MODELS

Speech recognition application consist of following sequence models: Rasta, Mlp and Decoder. In PC version, All environment parameters are fixed at present. Here is a short presentation of the parameter setting of these three models:

#**---RASTA** (*extract acoustic vectors)*

-w	*30*	frame-length; analysis window size
-s	*10*	frame-shift; window step size
-S	*8000*	sample-rate; Sampling frequency
-n	*12*	number of output parameters
-q	*2*	degree of delta calculation
-c	*17*	num of critical-band-like filters
-u	*29.021531*	upper cutoff zero freq
-m	*10*	model order
-p	*0.940000*	pole position
-r	*1.000000*	for partially rasta, partially plp
-W	*0.540000*	windowing constant
-l	*0.600000*	liftering exponent
-M		adds a small constant to the power spectrum
-d		debug
-L		for log rasta; log-rasta=yes

#**---MLP**(*evaluate likelihood of the phonemes)*

0	sort of use(0:reco; 1:training)
../SFP.234-600-36.softmax	file of weights and biases
234	nodes num of input layer
600	nodes num of hidden layer
36	nodes num of output layer

#**---DECODER**(*do the recognition)*

-dictionary ../data/diction.French
name of the file that contains vocabulary words with their transcriptions

-mlp_phonemes ../data/phonemes.French
name of the file with phonemes of MLP

-binary ../data/lm.bin
name of the file with language model in binary format

-states 2
number of states per phoneme

-silence
whether one wants to add '#h' (silence) as last phoneme to the vocabulary words

-alpha 0.25
scaling factor (additioner, will be logarithmed)

-beta 6
scaling factor (multiplier)

APPENDIX B

***VITERBI Recognition Results** (before optimization)*

message00.au   running time on Unix:6.55 running time on PC:4.407

aller au bonjour pour les ce jours que de n'ai la ce d'abord restaurant italien en pour centre de martigny la mais par cherche

message01.au   running time on Unix:10.22 running time on PC:6.672

c'est ouvert le je voudrais sais ce moi pour parce faire que cette je □□□□□□□□□□□□□□□□□□□□□Й□□О□□□□□□□ □□□И□□□□□□□

message02.au   running time on Unix:6.46 running time on PC:3.375

□□□□□□□□ □□□□□ □□□□□□□□□□□ □□□ □□□□□□□□□ □□□ □□□ □□□□□□□ □□И□□ □□□ □□□□□□□□□ famille avec des prix raisonnable

message03.au   running time on Unix:4.78 running time on PC:2.906

le mexicain restaurant du italien en adresses merci donc centre de ville martigny si

message04.au   running time on Unix:3.85 running time on PC:2.500

c'est soixante quand restaurant on parce aimerait que au-□□□а□ □□□□□□□□ soixante

message05.au   running time on Unix:8.23 running time on PC:5.843

□□□□□□□□□ а□ □□□ □□□□□ □□□□□□□□□□□ □□□О□□ □□□□□ □□□ □□□□□□□□□□ □□□□□□□ □□□□□ а□ □□□□□□□□□□□□ □□□□□□ □□□□□□□□□□□ □□□□□□□ □□□□ □□□□ □□□□ □□□□□□□□ □□□ □□И□□ restauration pas dire compris exactement oui cent pas merci bourg

message06.au   running time on Unix:3.56 running time on PC:2.781

c'est jusqu'а□□

message07.au   running time on Unix:7.59 running time on PC:4.125

□□□а□□□□□Й□□□□□□- □а□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□О□□□□□□О□□□□□□□□□□□□□

message08.au   running time on Unix:1.92 running time on PC:1.578

ou le macdo