TR-IT-0223

Japanese Phoneme Recognition Experiments on ATR's Travel Task (SDB) using HTK v2.02

コンラツド サンンダーソンン
Conrad Sanderson

シンガー ハラルド
Harald Singer

1997.05

This report describes the usage of HTK v2.02 software on the SDB database for the purposes of creating and testing context-independent and context-dependent acoustic models. Recognition accuracies are compared with the ATR SPREC recognizer. Structure of using HTK software, bugs and other particularities are presented. This report is a companion to TR-IT-0206.

Contents **Page**

1.0 Introduction

The purposes of this report are:

i) to give an introduction and examples of why and how to use HTK.
ii) to compare the performance of ATR's recognizer with that of HTK v2.0
iii) show problems encountered when using HTK on the SDB database

This report is not meant to be a replacement for HTK documentation [1] and should not be in any way interpreted as such. It merely provides a starting point for the usage of HTK.

This document is a companion document to TR-IT-0206, by Harald Singer et al [2].

The reader is encouraged to refer to the HTK manual whenever a more thorough explanation is required, e.g. the exact description of the purpose of a HTK tool.

In this report, wherever HTK is mentioned it refers to HTK version 2.0, unless explicitly stated otherwise.

HTK stands for Hidden Markov Model Toolkit and has been developed by Steve Young, Joop Jansen, Julian Odell, Dave Ollason and Phil Woodland. It is officially maintained by Entropic Research Laboratories.

1.1 Why Use HTK ?

Why should researchers at ATR use HTK while the ATRSPREC recognizer is available?

- visiting researchers may already be familiar with HTK
- many authors of technical papers and speech recognition literature use HTK as their basic system for speech recognition research
- the performance of the ATRSPREC recognizer can be compared with HTK when the same database is used
- the recognition transcription files from both recognizers can be compared to see where each one makes mistakes, thus indicating areas where the ATR recognizer can be improved or is better than other recognizers
- HTK is a commercial product with a reference manual, source code, technical support and frequent[1] updates.

1.2 Caveat about HTK

Before using HTK v2.0, please make sure that at least the v2.02 patch has been applied. The original HTK v2.0 software contains many bugs which can produce unexpected crashes or incorrect results.

The v2.02 patch is stored in **/dept1/work1/V1/htk/v2patch/**

1.3 Path for HTK scripts

The script package developed by the authors is stored in the **/dept1/work1/V1/htk/** directory. The user is encouraged to copy this directory into their own home directory before using the scripts.

[1] this is mainly due to the number of bugs that are continuously being discovered

2.0 HMM Based Recognition System Basics

Before getting into explanation of HTK workings, basics of a HMM based speech recognition system are shown in figures 1a to 1d.

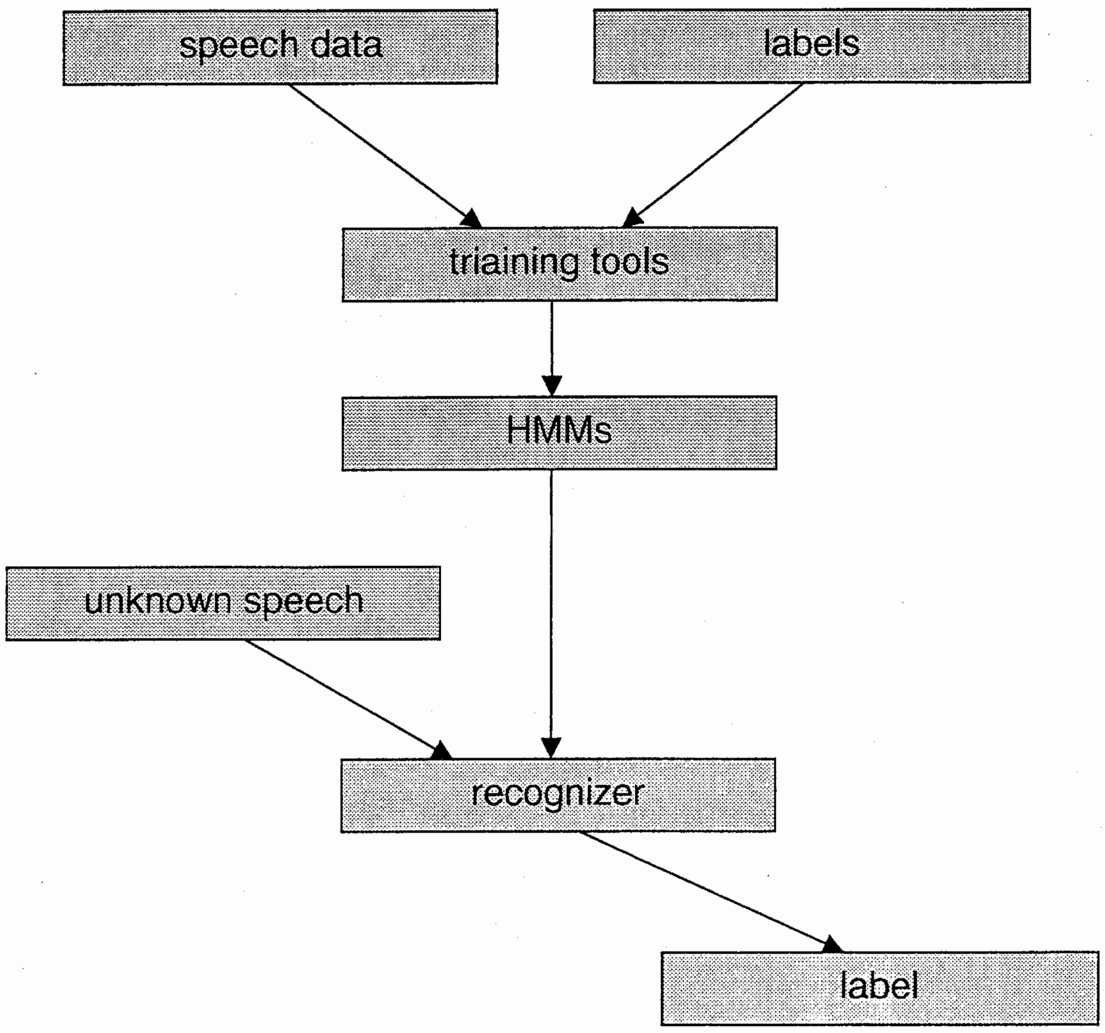


Figure 1a: basics of a HMM speech recognition system.

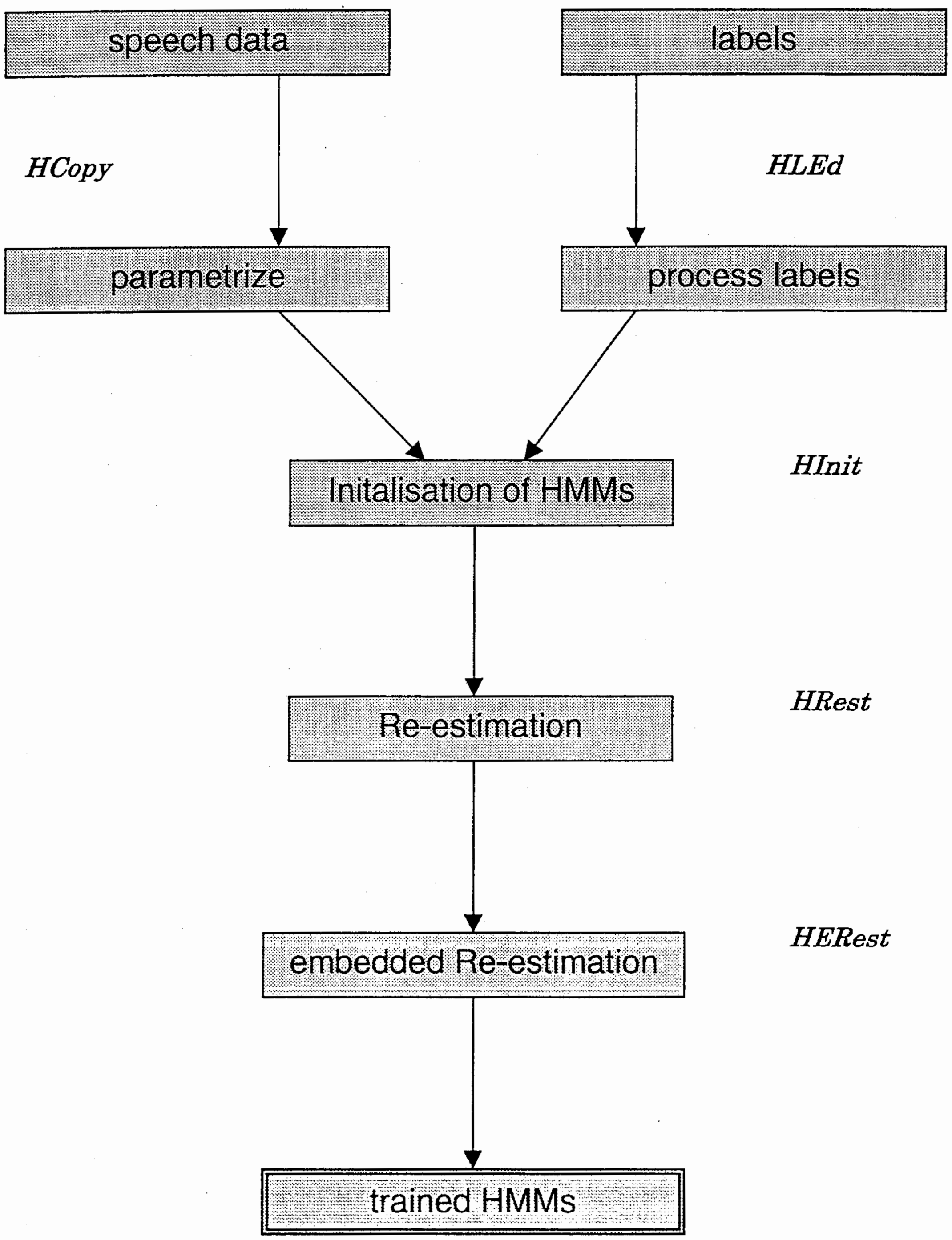


Figure 1b: basic training procedure using HTK.
The names in italics represent the HTK tools used.

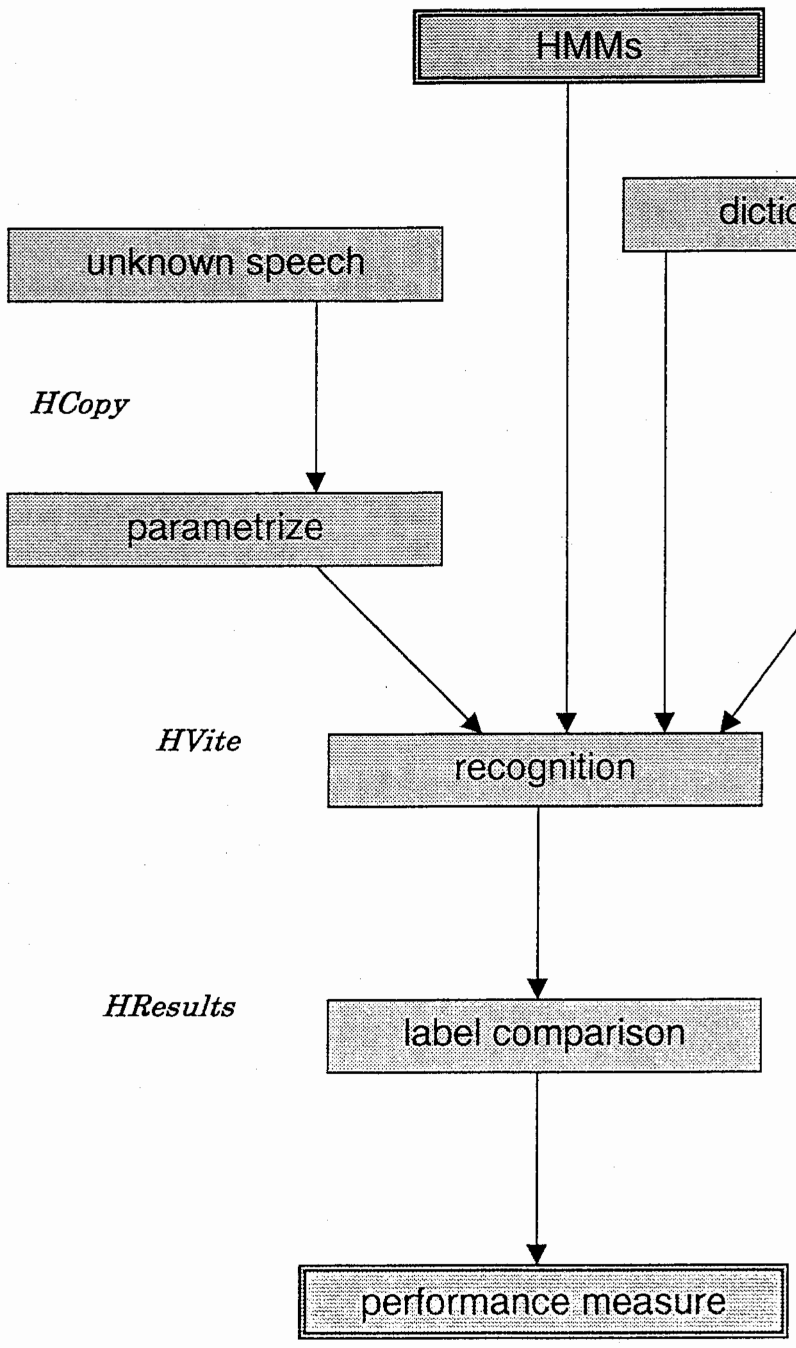


Figure 1c: basic recognition and performance evaluation procedure using HTK.
The names in italics represent the HTK tools used.

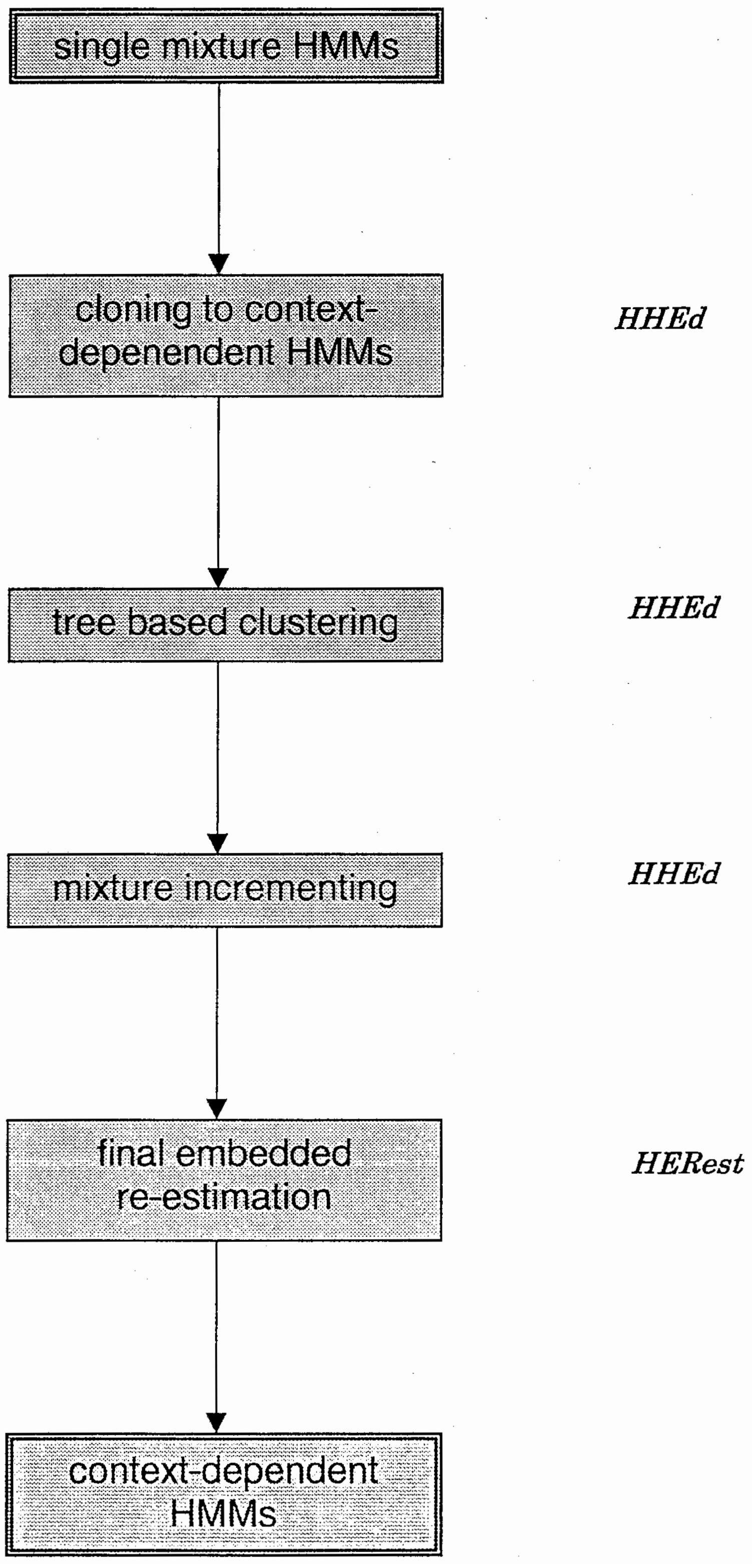


Figure 1d: basic training procedure for context-dependent HMMs using HTK.
The names in italics represent the HTK tools used.

2.1 Overview of HTK

HTK is made up of many programs, each with a dedicated task. These programs are used in a sequence in order to train and test acoustic models. Below is a table of the programs used by the authors in this report with a brief description of their function. This is by no means the entire list of HTK programs - merely the most used ones.

Program	Function
HCopy	Parameterize wave files (e.g. using LPC, MFCC).
HLEd	Edit label files (e.g. insert silences, convert to triphones).
HInit	Initialise a HMM model given a prototype HMM definition using Viterbi alignment (single mixture case) or K-means clustering (multiple mixture case).
HRest	Re-estimate a HMM model using Baum-Welch re-estimation.
HERest	Embedded Baum-Welch re-estimation of a set of HMM models.
HHed	Edit HMM definitions (e.g. increase number of mixtures, tree based clustering).
HParse	Convert a simple network definition into SLF network.
HVite	Perform recognition or alignment. Results of recognition are stored as label files.
HResults	Compare one set of label files to another.

To use these programs effectively and efficiently, a set of C shell and Python scripts as well as two C programs have been developed by the authors. Usage of these scripts is categorized into 3 stages, as shown in figure 2. In later sections of this report, each stage will be explained.

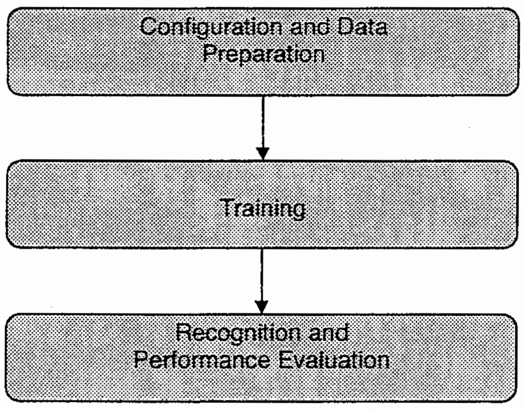

Configuration and Data Preparation
Training
Recognition and Performance Evaluation

Figure 2: flow diagram for using HTK scripts

3.0 C shell scripts

The script package has to be installed in its own directory (i.e. *sdbhtk*). There are 6 types of sub-directories used by the scripts:

Directory	Purpose
bin	Contains C source code and executables for programs which support the scripts
scripts	Contains all of the C shell scripts
lists	Contains lists used and generated by the scripts. A "list" can be a file containing a list of phones, transcriptions (labels), path of parametrized files, etc. An example: *train.monolabels.mlf* contains the labels for context independent training
hmm.x	Contains HMM definitions and related files. The "x" denotes the training level – see figure 3 for more information. The scripts will create these directories as necessary.
train	A link to the train directory from which training data is to be used. For more information about the SDB database, see the section 4.0
test	A link to the test directory, from which testing data is to be used during recognition

The C shell scripts have been written in such a way as to be configurable without direct modification. The main directory is left empty for the user to store files such as logs or notes. The only exception is the configuration file, named *scriptconfig*, where parameters needed for the execution of the scripts are contained. These parameters can be modified by the user. The *scriptconfig* file contains comments to aid the reader.

The scripts package is provided *as is*. The scripts have been tested and to the best of the authors' knowledge they work as they should. However, not all situations have been tested and hence no guarantees are provided. The scripts are not perfect and can probably be improved in many areas.

Note: whenever this report mentions a file used/generated by the C shell scripts (e.g. in flow diagram figures), it has been assumed that its stored in the *lists* directory, and the *lists/* prefix is **omitted.**

The reader is encouraged to peruse each script in order to get familiar with the scripts.

For more information about C shell scripts (in general), please see [6].

3.1 Python Scripts

In addition to the C shell scripts, three Python scripts have been developed. They are:

script name	purpose
`SSS2HTKpara.py`	parametrize files using ATR SPREC software and convert them to HTK compataible format
`SSS2HTKres.py`	converts ATRSPREC results into HTK label files for scoring with HTK software
`SetupJapaneseHTK.py`	reads in test and training SDB description files and: i) creates a directory structure ii) makes symbolic links to 16 kHz wave files iii) symbolic link to .SSS files and converts to .phn files (HTK compataible label format) iv) converts .TRS files to .phn files (without time information) if no .SSS file exists

and are stored in "`/dept1/work1/V1/htk/scripts/python/`". `SetupJapaneseHTK.py` needs to be run before any of the HTK experiments can be performed on the SDB database.

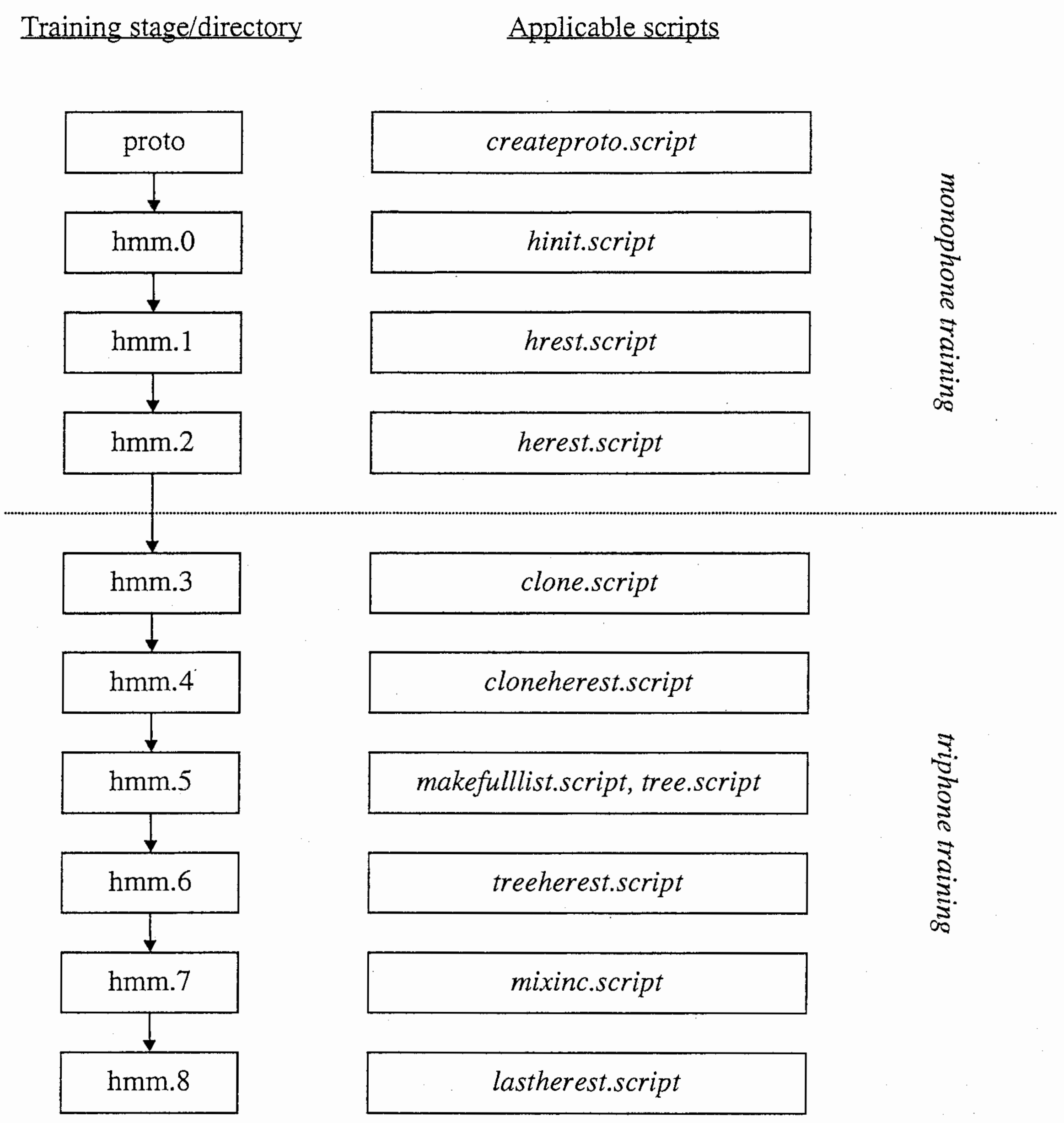
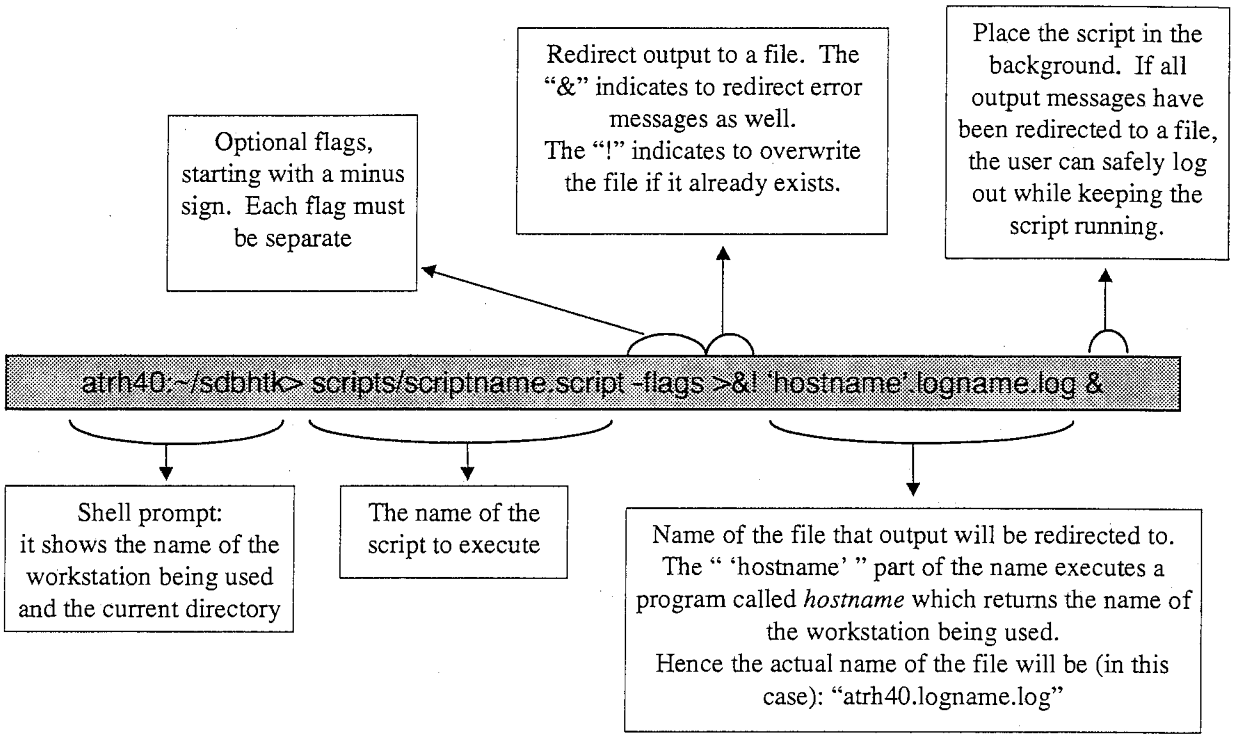


Figure 3: training directories/stages and applicable scripts

3.2 Using the C shell scripts

To use the scripts, the current directory must be the HTK script package directory. If using the C shell as the default shell, the scripts are executed in the following manner:



```
atrh40:~/sdbhtk> scripts/scriptname.script -flags >&! `hostname`.logname.log &
```

Optional flags, starting with a minus sign. Each flag must be separate

Redirect output to a file. The "&" indicates to redirect error messages as well. The "!" indicates to overwrite the file if it already exists.

Place the script in the background. If all output messages have been redirected to a file, the user can safely log out while keeping the script running.

Shell prompt: it shows the name of the workstation being used and the current directory

The name of the script to execute

Name of the file that output will be redirected to. The " 'hostname' " part of the name executes a program called *hostname* which returns the name of the workstation being used. Hence the actual name of the file will be (in this case): "atrh40.logname.log"

The main reason for using logfiles is that the scripts can be started on a different computer (e.g. a faster computer or a computer with more memory) and the output can be viewed on the machine currently being used Also, a record is kept of the progress of the scripts and any warnings and/or errors which could otherwise be overlooked.

3.3 Viewing logfiles

The log file can be viewed while it is being generated by a script using the "*tail –f*' command.

3.4 Flags for scripts

Flag	Function
-new	If the corresponding *hmm.x* directory contains a full set of HMMs, rename it to *hmm.x.old* and create a new *hmm.x*. Any previous *hmm.x.old* is deleted.
-wait	Wait until the directory *hmm.x-1* (i.e. preceding training directory) contains a full set of HMMs.

the *-new* flag applies to *hinit.script*. *–new* and *–wait* flags apply to *hrest.script*, and *herest.script* as well as *cloneherest.script*, *treeherest.script*, and *lastherest.script*.

3.5 Parallel mode in *hinit.script* and *hrest.script*

hinit.script and *hrest.script* can be executed in parallel on machines of the same architecture. This significantly reduces the time taken to train HMMs. The method to achieve this parallel execution is rather primitive so it has limitations.

The first invocation of the scripts can contain the flags described in section 3.3, however, each following invocation must not use any flags.

During monophone HMM training (upto *hrest.script*), each HMM can be stored as a separate file. Hence the same HTK tool (i.e. *HInit* or *HRest*) can be invoked in parallel, with each invocation working on a different HMM.

For each HMM to be trained, there exists a "*.lock*" file, e.g. "*a.lock*". Once the HMM is initialized or re-estimated, a "*.done*" file is created along with the HMM file (e.g. "*a.done*" and "*a*"). Once all of the HMMs have been trained, all of the ".done" files are deleted and a file named "*hmm.x.alldone*" is created, indicating that the complete set of HMMs has been processed for that directory.

3.6 Terminating the scripts

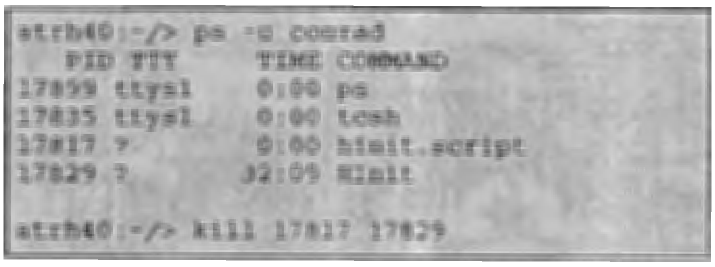
If for some reason a script needs to be terminated (e.g. incorrect input data) the user has to:

i) log onto the machine where the script is running

ii) find out the process numbers for the script AND any programs (the HTK tools) that the script has started. This is done using the "ps -u username" command, where "username" is the name of the user who started the scripts.

iii) kill the script FIRST, followed by killing any related HTK programs. The script has to be killed first otherwise the script will think that the HTK tool it started has failed and could, for example, start another copy of the tool for a different HMM (e.g. *hinit.script, hrest.script*)

The kill procedure for *hinit.script* is demonstrated below:



```
strb40:~/> ps -u conrad
   PID TTY      TIME COMMAND
17859 ttys1     0:00 ps
17835 ttys1     0:00 tcsh
17817 ?         0:00 hinit.script
17829 ?        32:09 HInit

strb40:~/> kill 17817 17829
```

3.7 Mail from Scripts

Since certain scripts take a long time to execute (e.g. *hinit.script, hrest.script, herest.script, hvite.script*), some scripts will send mail to the e-mail address specified in *scriptconfig*. (look for the *mailto* variable in *scriptconfig* and modify it so it is your e-mail address).

The message looks similar to the one below:

```
To: conrad@itl.atr.co.jp
From: Big Brother <bigbrother@atrh40.HQ.org>
Subject: scripts/hrest.script has finished
```

The message is addressed from *Big Brother* to easily distinguish it from all other e-mail you receive. There is no body to the message - the content of the message is embedded in the *Subject* field. This is done because most automatic mail checking programs display the subject when informing the user that new mail has arrived.

One other possible message is: "*hinit/hrest has failed on phone X*". This can be due to bugs present in HTK - sometimes the Viterbi alignment or K-Means clustering fails in the *HInit* tool. Otherwise the license server for HTK is not responding or is down.

4.0 Database

The Japanese non-read, "spontaneous" Speech Database was provided and has been organized into 2 directories: *train* and *test*. Each directory contains waveform files and their corresponding transcription files. The *train* directory contains 2962 waveform files while the *test* directory contains 551 waveform files. The source waveform files have no header and have been sampled at 16 kHz. Their byte-ordering is big-endian.

The transcription (label) files (*.sss) have been converted into HTK compatible format (*.phn) from their original format used by the ATR ML-SSS recognizer using the "`SetupJapaneseHTK.py`" script.

The transcriptions in the database are made up of 26 phones, as seen in figure 4.

phone	example
a	(赤い) akai
b	(便利) benri
ch	(力) chikara
d	(出口) deguchi
e	(円) eng
g	(技術) gizhjutsu
h	(旗) hata
i	(今) ima
j	(山) jama
k	(決定) keqtei
m	(万) mang
n	(名前) namae
ng	(電話) dengwa
o	(男) otoko
p	(パン) pang
q	(あっさり) aqsari
r	(論文) rongbung
s	(差) sa
sh	(処理) shjori
t	(谷) tani
ts	(使う) tsukau
u	(牛) ushi
w	(電話) dengwa
z	(安全) angzeng
zh	(味) azhi
sil	Pause or silence

Figure 4: names of phones, and examples, used in the SDB database

4.1 Bugs in Database

A "C" program called *biread.c* (stored in the *bin* directory) has been created by the authors. Its purpose is to read in a phone pair grammar matrix (stored as *lists/bigram.matrix*), which similar to the one shown in figure 5a (the blank spaces are filled with 0), and output a list of all possible triphones. It is used by *makefulllist.script* during the triphone HMM training procedure. The matrix is based upon the following files: `$ATRSPREC/etc/onso_setsuziku.type`
`$ATRSPREC/etc/phone_list.type`

	$	sil	a	i	u	e	o	k	s	sh	t	ch	ts	h	p	n	m	r	w	g	z	zh	d	b	j	q	ng
$	1		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
sil	1		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
a	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
i	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
u	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
e	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
o	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
k			1	1	1	1	1																		1		
s			1		1	1	1																				
sh				1																					1		
t			1			1	1																				
ch				1																					1		
ts					1																						
h			1	1	1	1	1																		1		
p			1	1	1	1	1																		1		
n			1	1	1	1	1																		1		
m			1	1	1	1	1																		1		
r			1	1	1	1	1																		1		
w			1																								
g			1	1	1	1	1																		1		
z			1		1	1	1																				
zh																									1		
d			1	1	1	1	1																				
b			1	1	1	1	1																		1		
j			1		1		1																				
q								1	1	1	1	1	1	1	1					1			1	1			
ng	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		

Figure 5a: phone pair grammar matrix. The left-most column lists all of the starting phones and the top row lists all of the ending phones. A "1" in the matrix indicates a valid bi-phone. The "$" symbol indicates start or end of a word or sentence.

biread outputs the list of triphones in HTK format. HTK representation of triphones is as follows:

(left context of phone) – (phone) + (right context of phone)

Figure 5b contains a list of triphones which are present in the training section of the database, but not present in the list of triphones which *biread* generated.

a-ch+e	t-i+i
a-ng+ng	u-ch+e
a-ng+q	u-ng+ng
a-t+i	u-t+i
ch-e+q	zh-e+e
e-ch+e	sil-ch+e
e-ng+q	sil-ng+d
e-t+i	sil-ng+g
i-ch+e	sil-ng+ng
i-ng+q	sil-ng+r
i-t+i	sil-ng+s
ng-ng+d	sil-ng+sil
ng-ng+sil	sil-ng+t
ng-ng+zh	sil-ng+ts
ng-q+t	sil-ng+zh
o-ch+e	sil-t+i
sil-zh+e	

Figure 5b: list of illegal triphones present in the training section of the SDB database

5.0 Parametrization

In order to compare the performance of HTK with that of SPREC, the same data and as similar as possible parametrization has to be used. Figure 6 shows the parametrization parameters used during experiments.

Parametrization Parameters	
Sample rate of wave files	12 kHz
Parametrization type	LPC derived Cepstral Coefficients
Observation vector	Cepstral Coefficients, energy, Δ cepstral coefficients, Δ energy
Number of cepstrum parameters	16
Linear Prediction analysis order	16
Tapering window	Hamming
Energy normalization	Yes
Cepstral mean subtraction	No
dimension of observation vector	34 (16 +1 + 16 + 1)
delta cepstrum computed over	± 4 frames
Pre-emphasis coefficient	0.98
frame length / size	20 msec
frame advance / shift	10 msec

Figure 6: parametrization parameters used during experiments

Notes:

- HTK uses a slightly different delta cepstrum computation than the ATR recognizer. SPREC computes delta cepstrum over a 9 frame triangular window (with the current frame being in the middle), while HTK uses a 9 frame rectangular window. See [7] and section 5.6 (pages 71-72) of [1] for more information.

- The ATRSPREC recognizer does not normalize the energy during parametrization

5.1 Problems with HCopy

The HTK tool HCopy incorrectly parametrizes files which contain a long string of zero amplitude samples.

The files in the SDB database which HCopy doesn't work with are listed in figure 7.

train/TAC70030/TAC700030.0190.B.12k
train/TCC70111/TCC70111.0100.A.12k
train/TCC70207/TCC70207.0180.A.12k
train/TCC70212/TCC70212.0110.B.12k
train/TCS70026/TCS70026.0160.A.12k
train/TCS70030/TCS70030.0200.A.12k
train/TCS70068/TCC70068.0080.A.12k
test/TAC70022/TAC70022.0240.A.12k
test/TCC70212/TCC70212.0040.A.12k
test/TCS70004/TCS70004.0030.B.12k
test/TCS70034/TCS70034.0280.A.12k

Figure 7: list of files which contain a long string of zero amplitude samples

There are 3 ways to solve this problem:

1) since the amount of files is negligible compared to the total size of the database, they can be ignored during training and testing

2) Find the offending bug(s) in HCopy and fix it.

3) add a very small amount of noise to the wave files before parametrization.

The authors' have chosen option 1, but a work around solution (3) has also been provided.

Option 2 would have required thorough understanding of HTK internals and the source code is not easily readable. It would have taken a significant amount of time to do this. Instead, the bug has been reported to Entropic Support.

The counterpart of *HCopy* in the HTK v1.5 distribution is called *HCode*. It parametrizes files in almost the exactly same way. For the differences, see page 67, footnote 2, page 71, footnotes 5 and 6, and page 72 of [1].

HCode v1.5 has been modified to add white Gaussian (zero mean and unit variance) noise to the wave files before parametrization. The SNR can be set by the user. By adding a very small amount of noise, the long strings of zeros are effectively corrupted without any significant corruption of the actual speech signal. Please see the **/dept1/work1/V1/htk/bin/** directory for the source code of the modified *HCode* tool.

The modified *HCode* tool may be useful for other purposes, such as testing the effects of varying amount of noise on recognition performance.

6.0 Data Preparation

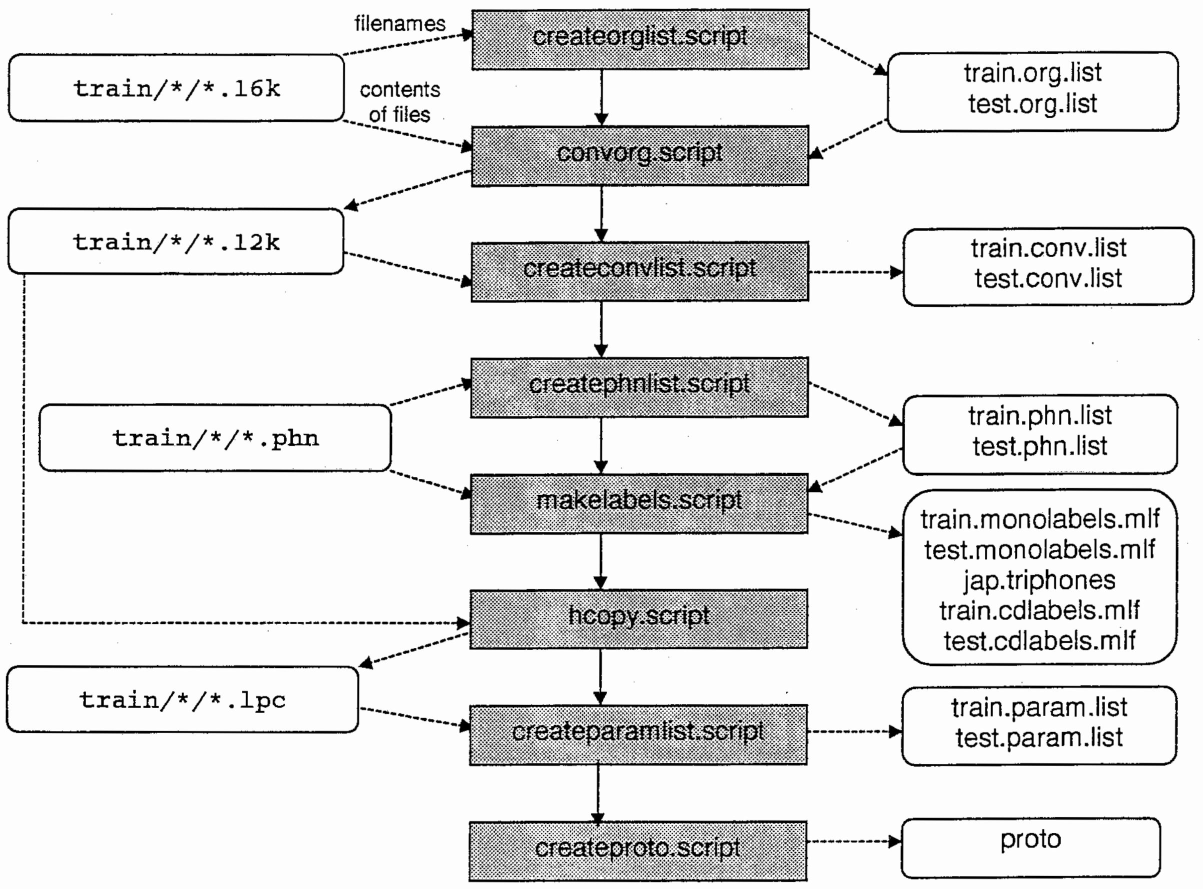


Figure 8: flow diagram of the data preparation stage

Data preparation is done as follows: (please refer to the figure 8 at the same time)

1) A list of wave files to be parametrized is obtained using *createorglist.script*. The script generates 2 lists: *train.org.list* and *test.org.list* corresponding to the original wave files (*.16k) in *train* and *test* directories. At the end of each list generation, a line count is performed for each list file, indicating the number of files. This provides a primitive check of the integrity of the database.

2) *convorg.script* is executed which downsamples the wave files from 16 kHz to 12 kHz. For each *.16k* file a *.12k* is generated and stored in the same directory.

3) *createconvlist.script* is executed to find all of the downsampled files. It works in a similar way to *createorglist.script* but it generates *train.conv.list* and *test.conv.list*. The reason another file list is generated (and another script is used) is to make sure that files for which downsampling failed are not processed from now on. A line count is performed for each list file, thus providing another check – i.e. if the number of files doesn't match the number of files reported in step 1, an error has occured during downsampling (e.g. due to lack of disk space).

4) *createphnlist.script* is executed to create a list of phonetic transcription files (*.phn). The lists are stored in *train.phn.list* and *test.phn.list*

5) *makelabels.script* is executed to create new label files where all of the phonetic labels are sorted according to the time sequence and any unlabelled segments with duration greater than 5 ms are marked as *sil* (i.e. silence). The output of this script are 2 files: *train.monolabels.mlf* and *test.monolabels.mlf.* The *mlf* stands for Master Label File and it contains all of the newly generated label files stored in one "master" file. It can be examined with a text editor. Please see section 6.3 (pages 95 – 99) in [1] for more information about MLFs.

Usage of MLFs is advantageous since the transcriptions are kept in one place instead of being scattered in the SDB directories. It also allows many experiments to be run concurrently without getting confused which label files belong to which experiment.

At the same time, triphone labels are generated, for later usage during triphone HMM training. A list of all the generated triphone names is saved to *jap.triphones.*

6) *hcopy.script* is executed to parametrize all of the downsampled files. For each *.12k* file, a *.lpc* file is generated and stored in the same directory.

7) *createparamlist.script* is executed which generates a list of all of the parametrized files. The lists are stored in *train.param.list* and *test.param.list.* This is done because HCopy, the HTK tool used by *hcopy.script*, contains bugs and can fail to parametrize a file.

8) *createproto.script* is used to generate a prototype HMM definition (stored as *proto*). The script sets up a prototype topology[2] definition for a 3 state, left to right HMM, with *user defined*[3] number of mixtures, 34 dimensional observation vectors. For the *sil* segment (silence) *proto.sil* is used by *hinit.script* and has to be manually created.

In the archive for the SDBHTK script package, *proto.sil* exists and is defined to be a single state HMM. The user is encouraged to view the *proto* and *proto.sil* files to get familiar with the structure of HMM definitions in HTK. Please see also section 7.2 (p 107 – 110) in [1] for more information about HMM definitions.

Please note that if the user wishes to train **context-dependent** HMMs (i.e. triphones), the number of mixtures has to be set to 1. The number of mixtures can be increased after triphone construction using a dedicated script.

[2] In HTK, a 3 state HMM is really a 5 state HMM, with the first and last states being dummy states.
[3] look for *numprotomixtures* variable in the *scriptconfig* file.

7.0 Training Context-independent (monophone) HMMs

This section will explain how to train context independent (i.e. monophone) HMMs

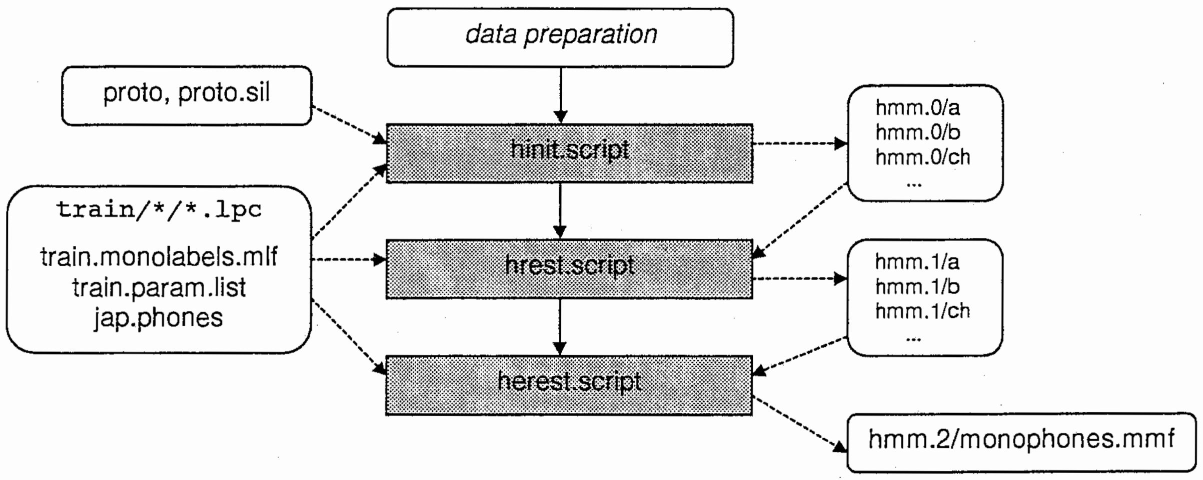


Figure 9: monophone HMM training

Please refer to figure 9 while reading this section.

1) Data preparation is performed as described in the previous section.

2) *hinit.script* is executed. The scripts requires a file called *jap.phones* containing the names of 26 Japanese phones, each corresponding to a HMM definition. Please see figure 3 for the list of the phones.

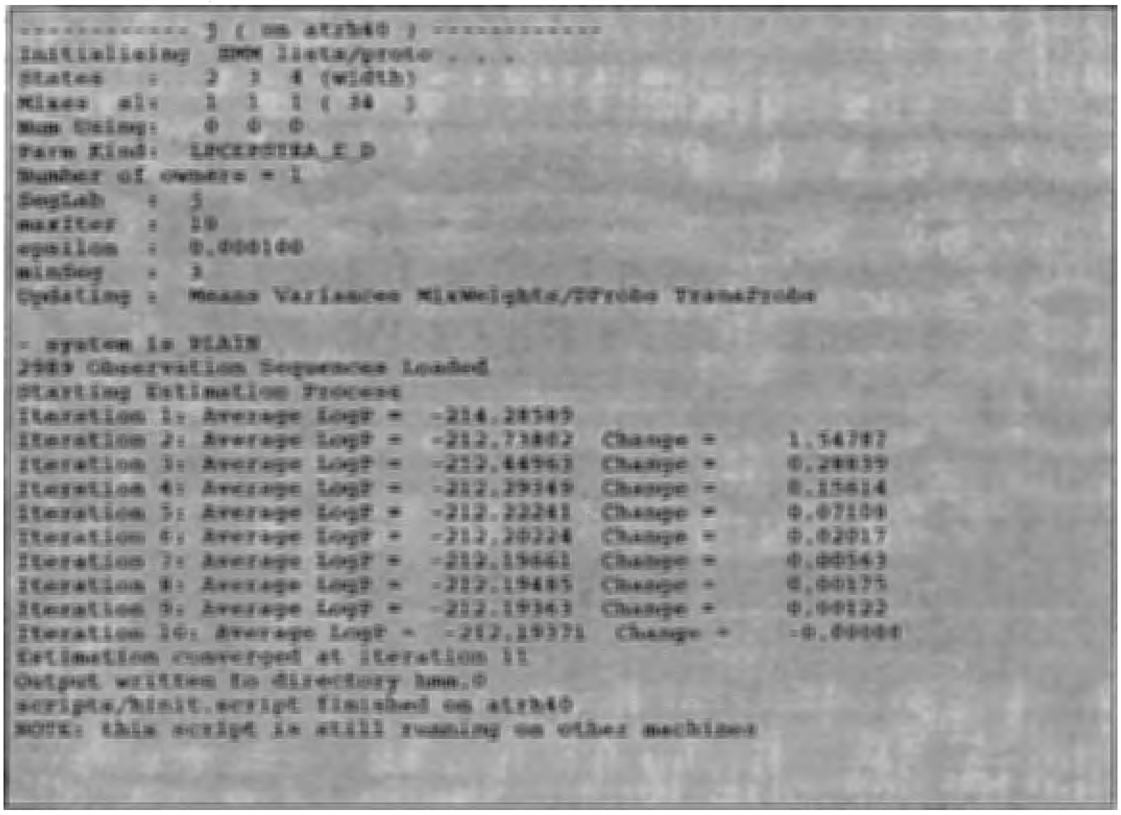
For each phone, *hinit.script* executes the *HInit* tool. *HInit* reads in the prototype topology HMM definition (*proto*) and provides initial estimates for the parameters of the HMM using the parametrized files stored in the *train* directory in conjunction with the *train.monolabels.mlf* master label file.

Note: for the phone "*sil*" the prototype "*proto.sil*" is used.

During initalization, *HInit* searches through all of the training files and reads in only the sections (observations) that correspond to the phone being initialized.

For single mixture HMMs, *HInit* starts by performing uniform segmentation of each observation. For multiple mixture HMMs, K-Means clustering is used to perform segmentation. *HInit* then uses Viterbi alignment to segment the training observations and then recomputes the HMM parameters.

The output of *hinit.script* looks similar to figure 10.



```
============== 3 ( on atrh40 ) ==============
Initialising  HMM lists/proto . . .
States   :   2  3  4 (width)
Mixes  s1:   1  1  1 ( 34  )
Num Using:   0  0  0
Parm Kind:  LPCEPSTRA_E_D
Number of owners = 1
SegLab   :  5
maxIter  :  20
epsilon  :  0.000100
minSeg   :  3
Updating :  Means Variances MixWeights/DProbs TransProbs

- system is PLAIN
2989 Observation Sequences Loaded
Starting Estimation Process
Iteration 1: Average LogP =  -214.28509
Iteration 2: Average LogP =  -212.73802  Change =     1.54707
Iteration 3: Average LogP =  -212.44963  Change =     0.28839
Iteration 4: Average LogP =  -212.29349  Change =     0.15614
Iteration 5: Average LogP =  -212.22241  Change =     0.07108
Iteration 6: Average LogP =  -212.20224  Change =     0.02017
Iteration 7: Average LogP =  -212.19661  Change =     0.00563
Iteration 8: Average LogP =  -212.19485  Change =     0.00175
Iteration 9: Average LogP =  -212.19363  Change =     0.00122
Iteration 10: Average LogP =  -212.19371  Change =    -0.00008
Estimation converged at iteration 11
Output written to directory hmm.0
scripts/hinit.script finished on atrh40
NOTE: this script is still running on other machines
```

Figure 10: typical output of *hinit.script*

3) *hrest.script* is executed. *hrest.script* reads the file *jap.phones*, containing the names of 26 Japanese phones, each corresponding to a HMM definition.

For each HMM, *hrest.script* executes the *HRest* tool, which reads in the previously initialized HMM from the *hmm.0* directory. Using the parametrized files stored in the *train* directory in conjunction with the *train.monolabels.mlf* master label file, Baum-Welch re-estimation is performed.

HRest searches through all of the training files and reads in only the sections (observations) that correspond to the phone being initialised.

4) execute *herest.script.* The script first combines all of the individual HMMs stored in *hmm.1* to a *Master Model File* (MMF) using the *HHEd* tool.

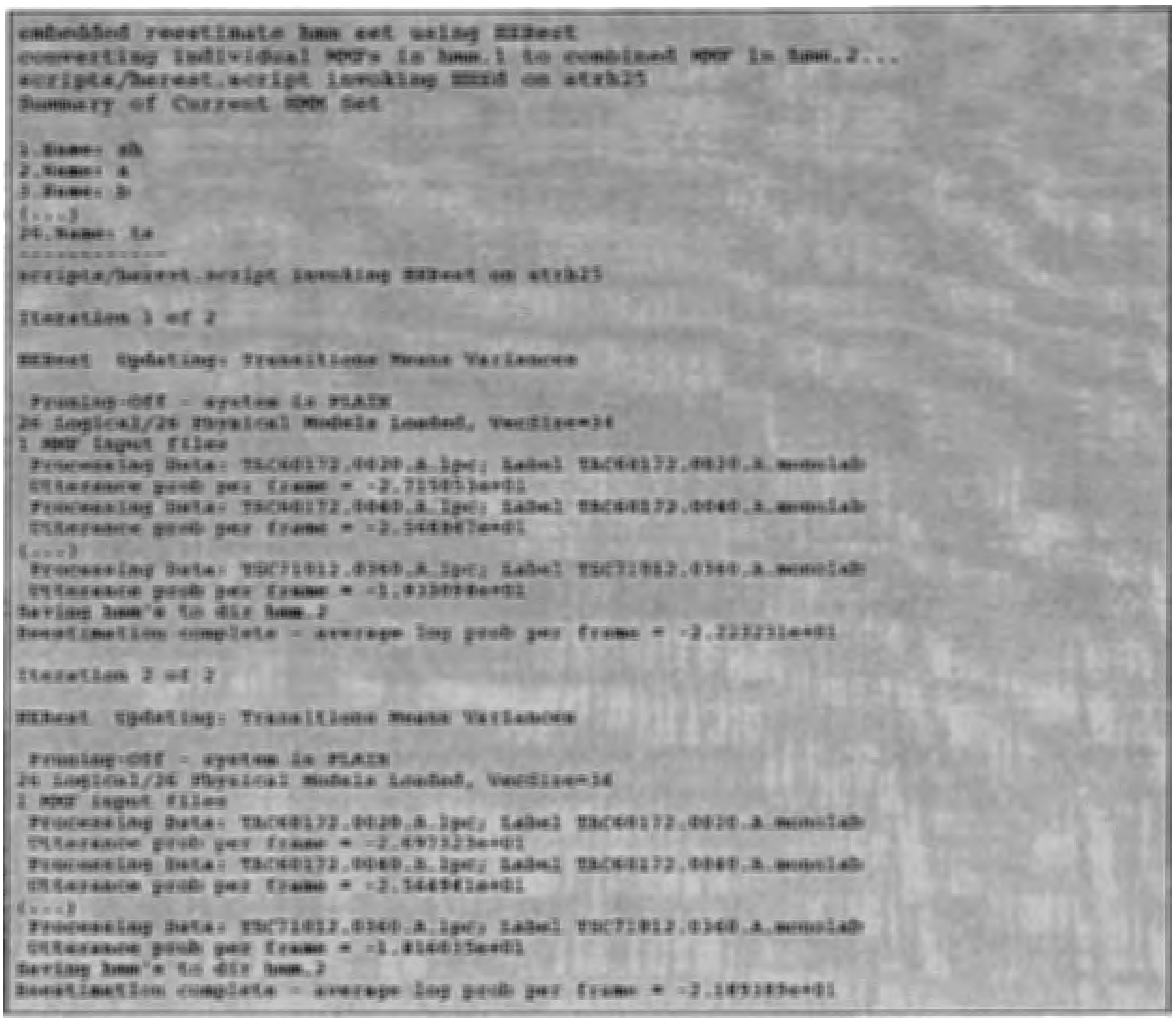
herest.script reads the *jap.phones* file, which contains the names of 26 Japanese phones, each corresponding to a HMM. The script then starts the *HERest* tool, which reads in the *MMF* and performs embedded re-estimation.

The *HERest* program performs a single pass of re-estimation of the parameters of the set of HMMs and the script.

herest.script runs *HERest* twice, hence 2 passes of re-estimation are done. The final set of HMMs is saved in *hmm.2/monophones.mmf.*

Note: *herest.script* can be only executed on one machine at a time - it does not work in parallel. An attempt has been made to write a script which uses the parallel feature of the HERest program, however the authors' could not get the script to work reliably. Please see section 8.5 (pages 134 to 137) in [1] for more information about the parallel mode of *HERest.*

The output of *herest.script* looks similar to figure 11.



```
embedded reestimate hmm set using HERest
converting individual MMFs in hmm.1 to combined MMF in hmm.2...
scripts/herest.script invoking HHEd on atrk25
Summary of Current HMM Set

1.Name: sh
2.Name: a
3.Name: b
(...)
26.Name: la
=============
scripts/herest.script invoking HERest on atrk25

Iteration 1 of 2

HERest  Updating: Transitions Means Variances

 Pruning-Off - system is PLAIN
26 Logical/26 Physical Models Loaded, VecSize=34
1 MMF input files
 Processing Data: TSC60172.0020.A.lpc; Label TSC60172.0020.A.monolab
 Utterance prob per frame = -2.715053e+01
 Processing Data: TSC60172.0040.A.lpc; Label TSC60172.0040.A.monolab
 Utterance prob per frame = -2.544867e+01
(...)
 Processing Data: TSC71012.0360.A.lpc; Label TSC71012.0360.A.monolab
 Utterance prob per frame = -1.813098e+01
Saving hmm's to dir hmm.2
Reestimation complete - average log prob per frame = -2.212231e+01

Iteration 2 of 2

HERest  Updating: Transitions Means Variances

 Pruning-Off - system is PLAIN
26 Logical/26 Physical Models Loaded, VecSize=34
1 MMF input files
 Processing Data: TSC60172.0020.A.lpc; Label TSC60172.0020.A.monolab
 Utterance prob per frame = -2.697323e+01
 Processing Data: TSC60172.0040.A.lpc; Label TSC60172.0040.A.monolab
 Utterance prob per frame = -2.544941e+01
(...)
 Processing Data: TSC71012.0360.A.lpc; Label TSC71012.0360.A.monolab
 Utterance prob per frame = -1.814635e+01
Saving hmm's to dir hmm.2
Reestimation complete - average log prob per frame = -2.185185e+01
```

Figure 11: shortened version of typical output of *herest.script*

8.0 Recognition using Monophone HMMs

This section explains how to perform recognition using trained monophone HMMs and then obtain performance measures.

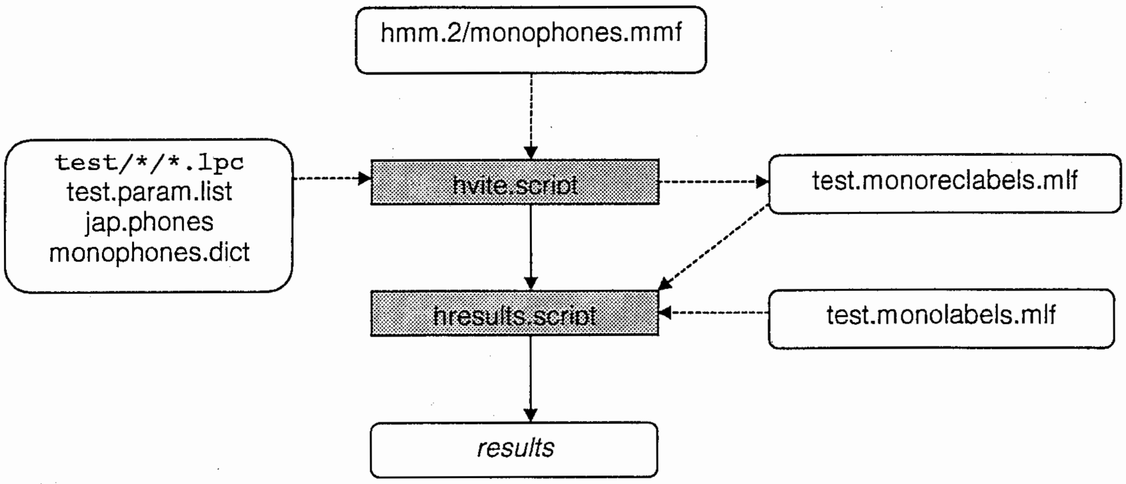


Figure 12: flow diagram for recognition (and performance evaluation) using monophone HMMs.

Please refer to figure 12 when reading this section.

1) *hvite.script* is executed. *hvite.script* uses the *HParse* tool to convert the HTK network definition file, *network.src*, into a Single Lattice File (SLF) used by *HVite*. The SLF is stored as *network*. Please see section 11.3 (pages 170 -172) in [1] for further explanation about network definitions.

The *HVite* tool is a general purpose *word* recognizer, and for our purposes we need to make it behave like a phone recognizer. A word dictionary has to be used. The file *monophones.dict* is a dictionary file and is used by *hvite.script.*

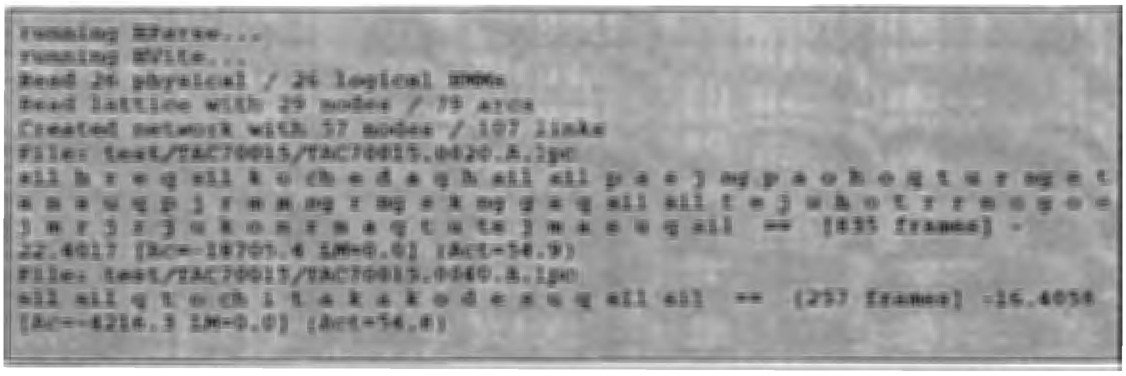
For every parametrized file to be recognized, HVite outputs a transcription file. *hvite.script* tells *HVite* to store all generated transcription files into one MLF, called *test.monoreclabels.mlf*

monophones.dict is used by *hvite.script.* The file is a dictionary describing the words and the phones used for each word. Since the purpose of this report is to compare the performance of HTK with ATRSPREC on the phone level, each "word" is the name of the corresponding phone, e.g.:

"word"	phone
a	a
b	b
ch	ch
...	...

Please see section 11.7, page 177, of [1] for more information about dictionaries.

The output of *hvite.script* looks similar to figure 13.



```
running HParse...
running HVite...
Read 26 physical / 26 logical HMMs
Read lattice with 29 nodes / 79 arcs
Created network with 57 nodes / 107 links
File: test/TAC70015/TAC70015.0020.A.lpc
all h r e q all k o ch e d a q h all all p a e j ng p a o h o q t u r ng e t
a n a u q p j r n n ng r ng e k ng g a q all all t e j u h o t r r n e g o e
j n r j r j u k o n r n a q t u ts j n a e u q all  ==  [835 frames] -
22.4017 [Ac=-18705.4 LM=0.0] (Act=54.9)
File: test/TAC70015/TAC70015.0040.A.lpc
all all q t o ch i t a k a k o d e s u q all all  ==  [257 frames] -16.4058
[Ac=-4216.3 LM=0.0] (Act=54.8)
```

Figure 13: typical output of hvite.script

2) *hresults.script* is executed. The tool *HResults* reads in the MLF generated by *HVite* as well as the MLF containing the correct transcriptions, (*test.monolabels.mlf*) and compares them. The comparison is based on a Dynamic Programming-based string alignment procedure. The output of *hresults.script* is shown in figure 18.

$$\% \text{ correct} = H / N * 100 \% = (N - D - S) / N * 100 \%$$

$$\% \text{ accuracy} = (H - I) / N \times 100 \% = (N - D - S - I) / N \times 100 \%$$

H	= hits
I	= insertions
D	= deletions
S	= substitutions
N	= total number of phones

9.0 Training Context Dependent (Triphone) HMMs

This section describes how to train 5 mixture, context dependent (triphone) HMMs.

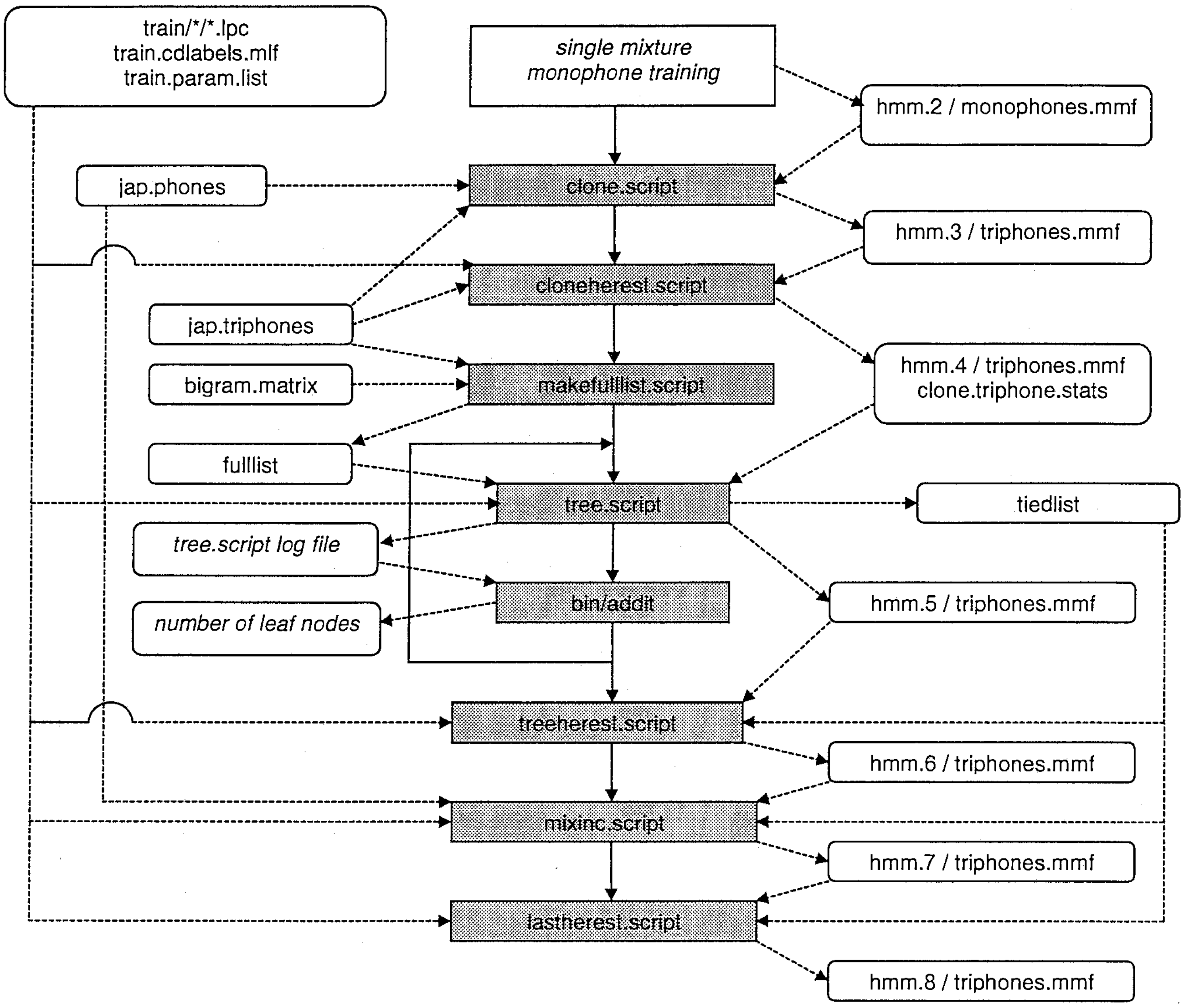
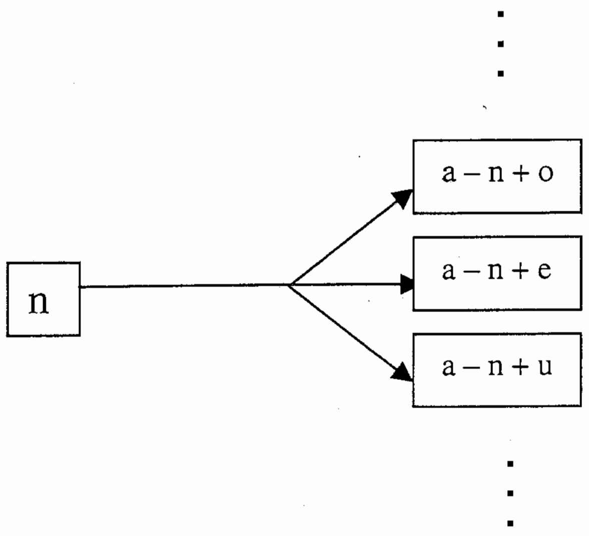


Figure 14: flow diagram for triphone HMM training

1) monophone training must first be performed - see section 7.0. A set of 1 mixture HMM models must be stored in *hmm.2/monophones.mmf*

2) *clone.script* is executed. This script clones all monophones into context-dependent versions (triphones). It obtains a list of triphone names from *jap.triphones*, which was generated during data preparation (section 6.0). For every triphone in *jap.triphones*, the corresponding monophone HMM is copied. e.g. the phone "*n*" is copied to "*a–n+o*", "*a–n+e*", "*a–n+u*", etc. All triphones with an "*n*" in the middle belong to the "*n*" family.



For the case of the context dependent phone "*a–n+o*", "*a–* " is the left context, while "*+o*" is the right context.

Once the triphones have been cloned, transition matrices for each triphone "family" are tied. A new MMF is created containing all of triphones and saved to *hmm.3/triphones.mmf.* The transition matrices need to be tied because the training data which would have been used for each of the original untied transitions is pooled so that a more reliable estimate can be obtained.

For more information about tying, see section 9.3 (pages 147 – 150) in [1].

3) The cloned triphones are re-estimated using *cloneherest.script.* 2 passes of embedded re-estimation are performed. At the end of the last pass, file *clone.triphone.stats* is generated, containing state occupation statistics. This file is required for state clustering, done in step 5.

4) *makefulllist.script* is executed to generate a list of all possible triphones present in Japanese. It uses the *biread* program which reads in a phone pair grammar matrix (see section 4.1 for more information) The list is merged with *jap.triphones* file (since the SDB database contains bugs in transcriptions). The merged list is saved as *fulllist.*

5) Tree based clustering is peroformed using *tree.script*

Why necessary is tree based clustering necessary ?

- context-dependent phones provide better recognition
- in Japanese, there is approximately 3000 context-dependent phones.
- No database contains sufficient amount of data to properly train them using a HMM system.
- Even if such database existed, the training time would take a very long time.

One solution to overcome the lack of training data is to share data during training. One method of sharing data is Tree Based Clustering. It has the advantage of being able to synthesise unseen (ie. not present in the training data) context-dependent phones. In HTK tree based clustering only works with single mixture, diagonal covariance HMMs. However, mixture incrementing is done afterwards.

A phonetic decision tree (binary tree) is constructed where each node has a yes/no **phonetic question**. Initally, all states (typically a specific phone state position) are placed in a single cluster at the root of the tree. A question is then found which gives the best split of the root node.

Each split of a cluster will increase the log likelihood since twice as many parameters are used to model the same amount of data.

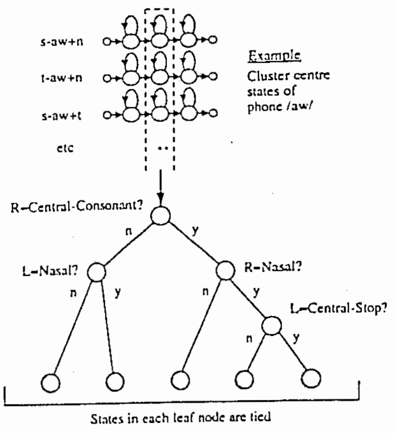
Hence the question which provides the most increase in log likelihood is selected.

The clusters are continously split until the states have trickled down to leaf nodes. All states in the same leaf node are tied - ie. they share the same data.

Splitting is also stopped when the increase in log likelihood falls below a specified threshold.

Example of tree based clustering:

The centre state of "**s – aw + n**" would join the second leaf node from the right since its right context is a central consonant, and its right context is a nasal but its left context is not a central stop.

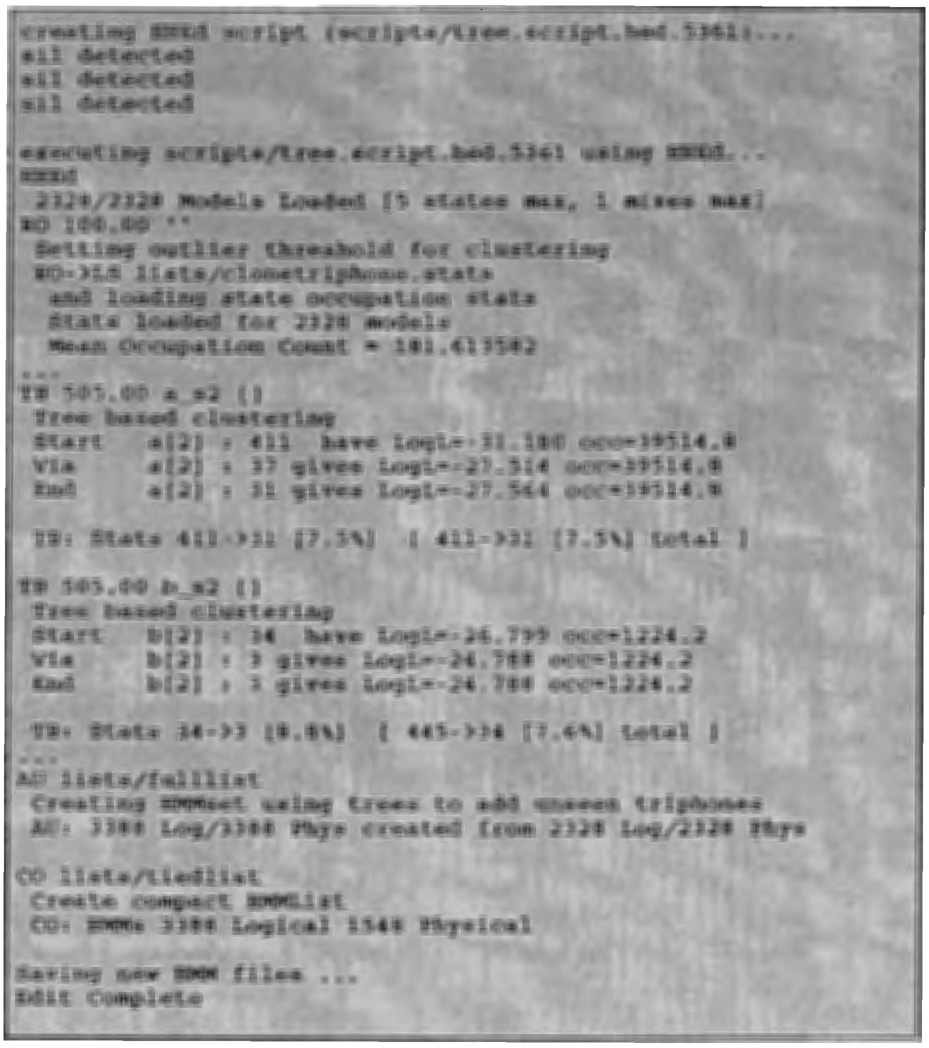


Please see [4] for more detailed explanation of tree-based clustering.

The list of phonetic questions applicable to Japanese has been obtained from [5] and have been converted into HTK HHEd format. The questions are stored as: "**lists/questions.hed**"

A new set of tied triphones is stored in *hmm.5/triphones.mmf.* A new list of tied triphones is saved to *tiedlist.*

The output of *tree.script* **has to be saved** to a log file so step 6 can be executed. The output of the script is similar to figure 15.



```
creating HHEd script (scripts/tree.script.hed.5361)...
sil detected
sil detected
sil detected

executing scripts/tree.script.hed.5361 using HHEd...
HHEd
 2328/2328 Models Loaded [5 states max, 1 mixes max]
RO 100.00 ''
 Setting outlier threshold for clustering
 RO->LS lists/clonetriphone.stats
  and loading state occupation stats
  Stats loaded for 2328 models
  Mean Occupation Count = 181.613582
...
TB 505.00 a_s2 {}
 Tree based clustering
 Start    a[2] : 411  have LogL=-31.180 occ=39514.8
 Via      a[2] : 37 gives LogL=-27.514 occ=39514.8
 End      a[2] : 31 gives LogL=-27.564 occ=39514.8

 TB: Stats 411->31 [7.5%]  { 411->31 [7.5%] total }

TB 505.00 b_s2 {}
 Tree based clustering
 Start    b[2] : 34  have LogL=-24.799 occ=1224.2
 Via      b[2] : 3 gives LogL=-24.788 occ=1224.2
 End      b[2] : 3 gives LogL=-24.788 occ=1224.2

 TB: Stats 34->3 [8.8%]  { 445->34 [7.6%] total }
...
AU lists/fulllist
 Creating HMMset using trees to add unseen triphones
 AU: 3384 Log/3384 Phys created from 2328 Log/2328 Phys

CO lists/tiedlist
 Create compact HMMList
 CO: HMMs 3384 Logical 1548 Physical

Saving new HMM files ...
Edit Complete
```

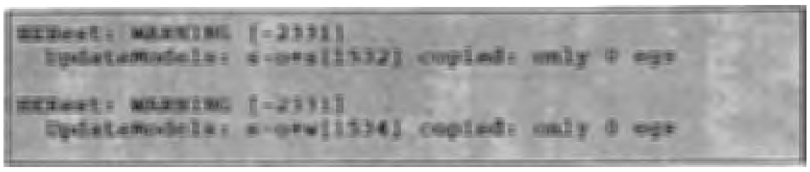
Figure 15: typical shortened output of *tree.script*

6) The logfile of *tree.script* is parsed by the program *addit*, which counts the total number of leaf nodes generated by tree based clustering. The number of leaf nodes generated is controlled by the *treetresh* variable in the *scriptconfig* file. It is set to 505.0, giving 810 leaf nodes, which is approximately the same as used in the ATRSPREC recognizer.

The *treetresh* variable can be changed to increase or decrease the number of leaf nodes. Steps 5 and 6 will need to be repeated until the desired number of nodes has been reached.

Example of using *addit*: "bin/addit tree.log 26 3". The 26 specifies the number of mono-phones and 3 specifies the number of states for each HMM.

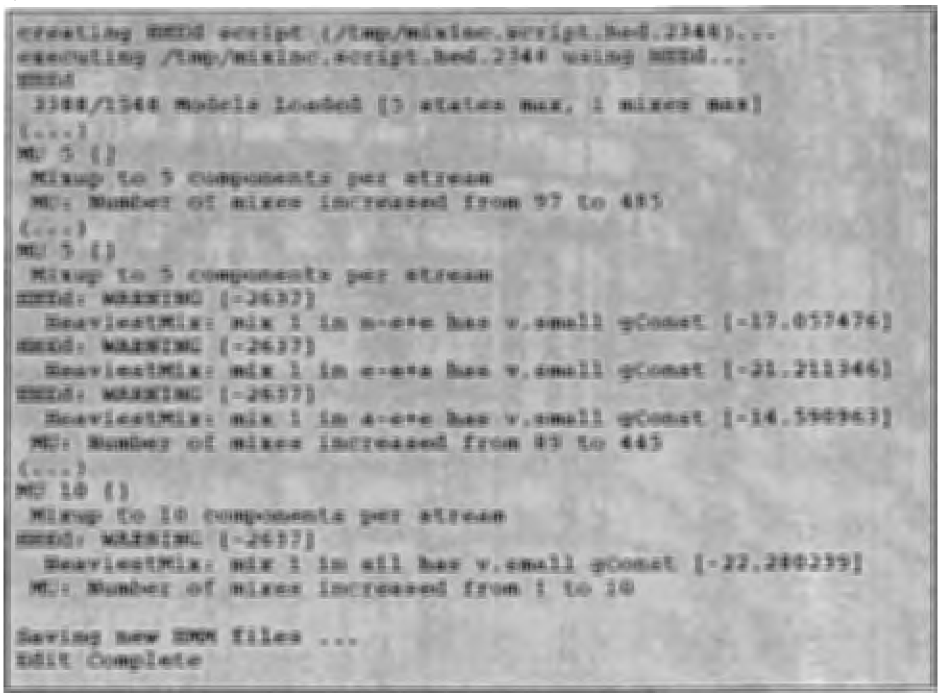
7) After tree based clustering, the new set of triphones has to be re-estimated. This is done with *treeherest.script*. It takes its input from *hmm.5/triphones.mmf* and performs 2 passes of embedded re-estimation. The re-estimated triphones are saved as *hmm.6/triphones.mmf*. During the re-estimation, the warnings similar to the ones shown below will occur. This is normal, since the triphone set now contains synthesised triphones (i.e. triphones which do not occur in the training data).



```
HERest: WARNING [-2331]
  UpdateModels: s-o+s[1532] copied: only 0 egs

HERest: WARNING [-2331]
  UpdateModels: s-o+w[1534] copied: only 0 egs
```

8) Mixture incrementing is done with *mixinc.script*. All triphone HMMs will have the number of mixtures increased from 1 to 5, with the exception of the phone *sil* which will have 10 mixtures. The new HMMs are saved as *hmm.7/triphones.mmf*. The output of *mixinc.script* is similar to figure 16.



```
creating HHEd script (/tmp/mixinc.script.hed.2344)...
executing /tmp/mixinc.script.hed.2344 using HHEd...
HHEd
 2344/2344 models loaded [3 states max, 1 mixes max]
(...)
MU 5 {}
 Mixup to 5 components per stream
 MU: Number of mixes increased from 97 to 485
(...)
MU 5 {}
 Mixup to 5 components per stream
HHEd: WARNING [-2637]
  HeaviestMix: mix 1 in m-e+e has v.small gConst [-17.057476]
HHEd: WARNING [-2637]
  HeaviestMix: mix 1 in e-e+a has v.small gConst [-21.211346]
HHEd: WARNING [-2637]
  HeaviestMix: mix 1 in a-e+e has v.small gConst [-14.590963]
 MU: Number of mixes increased from 89 to 445
(...)
MU 10 {}
 Mixup to 10 components per stream
HHEd: WARNING [-2637]
  HeaviestMix: mix 1 in sil has v.small gConst [-22.280239]
 MU: Number of mixes increased from 1 to 10

Saving new HMM files ...
HHEd Complete
```

Figure 16: typical shortened output of *mixinc.script*

9) Finally, 5 passes of embedded re-estimation are done with *lastherest.script*. The resulting HMMs are saved as *hmm.8/triphones.mmf*.

10.0 Recognition Using Context Dependent (triphone) HMMs

Recognition using context dependent (triphones) HMMs is straightforward and very similar to recognition using context-independent (monophone) HMMs. Please refer to figure 17 while reading this section.

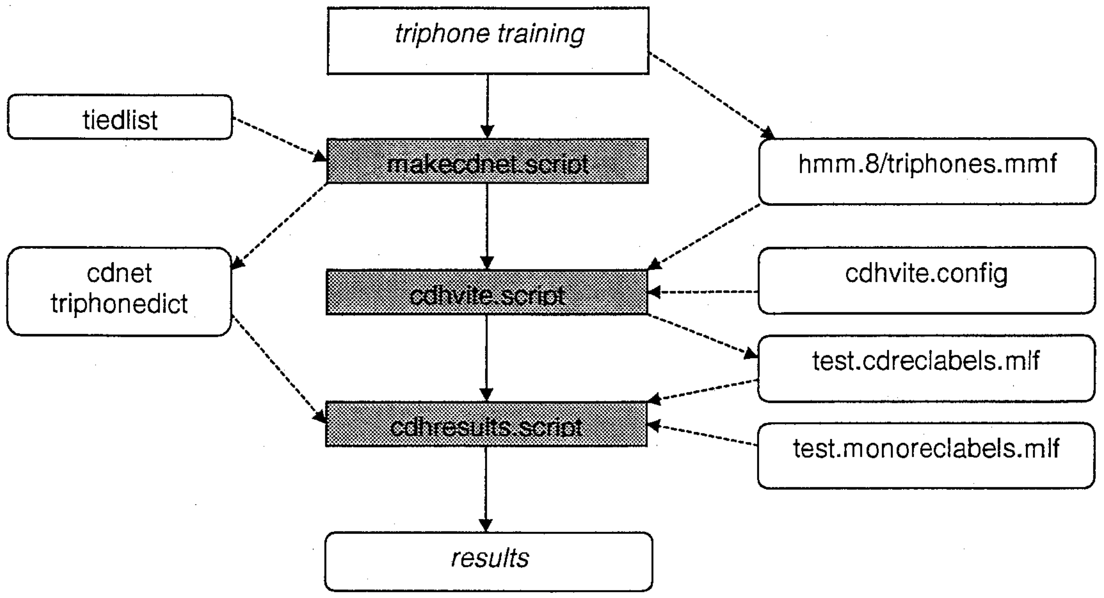


Figure 17: flow diagram for recognition (and performance evaluation) using triphone HMMs

1) triphone HMM training has to be performed – see section 9.0 of this report. The result of triphone training is stored in *hmm.8/triphones.mmf*, and *tiedlist* - both must exist before recognition can be performed.

2) *makecdnet.script* is executed. This creates a context-dependent network definition, using all of the triphones listed in *tiedlist.* The network definition is then processed by the tool *HParse* to convert it into a SLF. The SLF is saved as *cdnet* for usage by *cdhvite.script.* At the same time, a dictionary file called *triphonedict* is created (see section 8.0 of this report for more information).

The authors have found section 11.9 (page 184) of [1] to be incorrect. The authors have managed to get *HVite* working as a triphone recognizer only after reading section 4.5 (pages 50 – 53) of [3].

3) *cdhvite.script* is executed. It runs the *HVite* tool , which uses the configuration file *cdhvite.config*. For more information about the configuration variables, please see section 14.1 (pages 270) under the HNet module in [1].

4) Finally, *cdhresults.script* is executed to obtain performance measures. The output it generates is similar to figure 19. *HResults* will be run with options to strip phone contexts before comparing transcription files.

10.1 Obtaining performance figures for ATR recognizer using HTK software

For the purposes of obtaining comparison performance figures, it is necessary to use the same scoring software for both sets of recognition transcription files: one set belonging to HTK recognizer, and the other belonging to the ATR recognizer. The same scoring software has to be used since the DP matching algorithm may be slightly different and/or the equations used for the final accuracy are different.

atrcdhresults.script has been developed to compare the transcription files generated by ATR SPREC (provided in HTK format).

It assumes that the recognition transcription files are stored in "*/homes/singer/SDBHTK/test/*" and follow the structure of the SDB database. The script will generate two temporary *MLFs*, one from the given transcription files, and the second one from the MLF containing the original transcription files. This is necessary since *HResults*, the HTK scoring program, would get confused about the location of the transcription files.

The script requires the MLF containing the original transcription files to already exist. See section 6.0 (data preparation) for more information.

11.0 Results

Figures 18 contains recognition results on a per speaker basis using single mixture, context independent, (monophone) HMMs, on the test section of the SDB database.

HMM system	**number of gaussians**	**% correct**	**% accuracy**
single mixture, context-independent	76	55.63	43.02
15 mixture, context-independent	1135	70.78	60.99
5 mixture, context-dependent	~4000	83.48	67.99

For the 5 mixture CD case, triphone constraints were used during recognition.Figure 19 is similar to figure 18, except it shows recognition results using 5 mixture, context dependent (triphone) HMMs. Figure 20 is the counterpart to figure 19 – transcription files from the ATR recognizer have been converted into HTK format and the HTK scoring program, HResults, has processed them.

measurement	**HTK results**	**ATR SPREC results**	**difference wrt SPREC**
% correct	83.48	81.61	+ 1.87
% deletions	4.45	6.06	- 1.61
% substitutions	12.04	12.33	- 0.29
% insertions	15.49	10.17	+ 5.32
% accuracy	67.99	71.44	- 3.45

From the above results it can be seen that HTK software has a higher hit rate (1.87 % points) and a slightly lower deletions rate (1.6 % points) than the ATR SPREC recognizer. Substitutions rate for both recognizers is almost the same. However, the insertion rate in HTK software is significantly higher (5.32 % points). We could probably achieve even closer insertion and deletion rates by varying the word insertion penalty.

Figure 21 is similar to figure 18, except it shows the results for single mixture, context independent, 15 mixture HMMs.

Figure 22 shows results for context dependent, 5 mixture HMMs, using parametrized files processed with ATR SPREC software and converted into HTK format. During training, the log probabilities displayed by HTK tools were positive instead of the usual negative. The low rate of accuracy in the results shows that HTK is very sensitive to different parametrization. Due to time constraints we could not make the necessary adjustments to the software and/or find optimal threshold values.

For the context-independent results, no language model constraints were applied. For the context-dependent results, triphone constraints were applied. This makes a direct comparison between context-dependent and context-independent results impossible. Due to missing documentation and time constratins we could not figure out how to construct an equivalent grammer for the context-dependent experiments.

12.0 Conclusions

HTK and ATR SPREC gave very similar results for the context-dependent case which shows that ATR SPREC is close to the state of the art. To do a more precise component evaluation, e.g. tree-based clustering employed in HTK vs. MLSSS in ATR SPREC, we would need to reimplement the respective algorithms in the corresponding framework, e.g. the MLSSS algorithm in HTK software. Due to time constraints, this was not possible here.

We hope that the detailed explanation of procedures for training and testing context-independent and context-dependent HMMs using HTK will be useful for other researchers who have to do comparative evaluations. The acoustic models developed here might also be of use for automatic alignment of Japanese databases for speech synthesis purposes.

Unfortunately, Version 2.0 of HTK contains many bugs - even after applying patches. Entropic does not support v2.0 anymore and previous patches have been removed on their website. Upgrading to v2.1 should therefore be seriously considered.

```
========================= HTK Results Analysis ============================
  Date: Mon Mar 17 15:06:30 1997
  Ref : lists/test.monolabels.mlf
  Rec : lists/test.reclabels.mlf
------------------------------ Speaker Results -----------------------------
spkr: %Corr( %Acc )  [ Hits, Dels, Subs, Ins, #Words] %S.Corr [ #Sent ]
----------------------------------------------------------------------------
AC70015A:   58.89( 41.33)  [H= 275, D= 62, S=130, I= 82, N= 467]   0.00 [N= 14]
AC70016A:   46.94( 38.89)  [H= 169, D=100, S= 91, I= 29, N= 360]   0.00 [N= 12]
AC70017A:   50.39( 45.74)  [H= 130, D= 50, S= 78, I= 12, N= 258]   0.00 [N= 11]
AC70019A:   50.62( 40.25)  [H= 205, D= 78, S=122, I= 42, N= 405]   0.00 [N= 15]
AC70021A:   56.40( 44.49)  [H= 251, D= 47, S=147, I= 53, N= 445]   0.00 [N= 16]
AC70022A:   44.71( 32.40)  [H= 207, D=111, S=145, I= 57, N= 463]   0.00 [N= 12]
AC70023A:   57.97( 38.84)  [H= 291, D= 83, S=128, I= 96, N= 502]   0.00 [N= 17]
AC70101A:   61.85( 52.16)  [H= 287, D= 82, S= 95, I= 45, N= 464]   0.00 [N= 20]
AC70102A:   65.07( 46.14)  [H= 354, D= 56, S=134, I=103, N= 544]   0.00 [N= 18]
AC70103A:   45.99( 36.59)  [H= 132, D= 74, S= 81, I= 27, N= 287]   0.00 [N= 10]
AC70201A:   65.31( 56.20)  [H= 337, D= 74, S=105, I= 47, N= 516]   0.00 [N= 15]
AC70202A:   47.50( 39.93)  [H= 276, D=147, S=158, I= 44, N= 581]   0.00 [N= 11]
AC70203A:   60.48( 43.01)  [H= 329, D= 85, S=130, I= 95, N= 544]   0.00 [N= 18]
AC70301A:   60.26( 47.60)  [H= 276, D= 82, S=100, I= 58, N= 458]   0.00 [N= 11]
AC70303A:   65.97( 53.99)  [H= 314, D= 78, S= 84, I= 57, N= 476]   0.00 [N= 19]
AC70304A:   44.61( 27.88)  [H= 120, D= 33, S=116, I= 45, N= 269]   0.00 [N=  8]
CC70103A:   50.93( 38.93)  [H= 191, D= 74, S=110, I= 45, N= 375]   0.00 [N=  9]
CC70109A:   52.19( 37.53)  [H= 203, D= 84, S=102, I= 57, N= 389]   0.00 [N= 13]
CC70201A:   48.70( 37.39)  [H= 168, D= 85, S= 92, I= 39, N= 345]   0.00 [N=  7]
CC70212A:   47.39( 29.85)  [H= 254, D=114, S=168, I= 94, N= 536]   0.00 [N= 17]
CC70307A:   54.40( 42.31)  [H= 315, D=120, S=144, I= 70, N= 579]   0.00 [N= 11]
CC71001B:   51.60( 39.84)  [H= 452, D=197, S=227, I=103, N= 876]   0.00 [N= 12]
CC71007A:   63.61( 52.72)  [H= 409, D= 94, S=140, I= 70, N= 643]   0.00 [N= 26]
CC71008A:   48.22( 36.52)  [H= 338, D=144, S=219, I= 82, N= 701]   0.00 [N= 17]
CC71016A:   65.70( 46.35)  [H= 387, D= 73, S=129, I=114, N= 589]   0.00 [N= 16]
CC71035A:   63.80( 55.99)  [H= 245, D= 78, S= 61, I= 30, N= 384]   0.00 [N= 11]
CS70004B:   52.97( 43.84)  [H= 580, D=225, S=290, I=100, N=1095]   0.00 [N= 14]
CS70010A:   49.88( 40.90)  [H= 200, D= 95, S=106, I= 36, N= 401]   0.00 [N= 10]
CS70013A:   61.92( 50.12)  [H= 252, D= 66, S= 89, I= 48, N= 407]   0.00 [N=  7]
CS70020A:   56.31( 36.71)  [H= 250, D= 67, S=127, I= 87, N= 444]   0.00 [N= 11]
CS70023A:   58.02( 42.97)  [H= 532, D=139, S=246, I=138, N= 917]   0.00 [N= 14]
CS70025A:   53.56( 46.87)  [H= 248, D= 71, S=144, I= 31, N= 463]   0.00 [N=  9]
CS70028A:   63.56( 49.59)  [H= 232, D= 34, S= 99, I= 51, N= 365]   0.00 [N= 13]
CS70034A:   52.24( 38.34)  [H= 327, D=110, S=189, I= 87, N= 626]   0.00 [N= 17]
CS70047A:   62.86( 47.33)  [H= 259, D= 71, S= 82, I= 64, N= 412]   0.00 [N= 12]
CS70055A:   51.02( 40.70)  [H= 351, D=175, S=162, I= 71, N= 688]   0.00 [N= 11]
CS70059A:   59.62( 48.35)  [H= 217, D= 63, S= 84, I= 41, N= 364]   0.00 [N=  6]
CS70070A:   46.10( 33.44)  [H= 142, D= 67, S= 99, I= 39, N= 308]   0.00 [N=  8]
CS70074A:   57.28( 43.04)  [H= 181, D= 69, S= 66, I= 45, N= 316]   0.00 [N= 12]
CS70082A:   61.04( 48.31)  [H= 398, D=106, S=148, I= 83, N= 652]   0.00 [N=  8]
SC71005B:   54.84( 42.78)  [H= 300, D= 87, S=160, I= 66, N= 547]   0.00 [N= 15]
SC71013A:   55.56( 43.22)  [H= 500, D=160, S=240, I=111, N= 900]   0.00 [N= 14]
------------------------------ Overall Results -----------------------------
SENT: %Correct=0.00 [H=0, S=547, N=547]
PHONE: %Corr=55.63, Acc=43.02 [H=11884, D=3910, S=5567, I=2694, N=21361]
============================================================================
```

Figure 18: results of testing context independent, single mixture HMMs

```
========================== HTK Results Analysis ============================
  Date: Sun Mar 23 19:35:06 1997
  Ref : lists/test.monolabels.mlf
  Rec : lists/test.cdreclabels.mlf
----------------------------- Speaker Results -------------------------------
spkr: %Corr( %Acc )  [ Hits, Dels, Subs, Ins, #Words] %S.Corr [ #Sent ]
-----------------------------------------------------------------------------
AC70015A:  77.09( 55.03)  [H= 360, D= 26, S= 81, I=103, N= 467]   0.00 [N= 14]
AC70016A:  68.33( 55.00)  [H= 246, D= 33, S= 81, I= 48, N= 360]   8.33 [N= 12]
AC70017A:  82.95( 77.52)  [H= 214, D=  4, S= 40, I= 14, N= 258]   0.00 [N= 11]
AC70019A:  85.93( 75.31)  [H= 348, D= 12, S= 45, I= 43, N= 405]   6.67 [N= 15]
AC70021A:  80.45( 53.03)  [H= 358, D= 12, S= 75, I=122, N= 445]   0.00 [N= 16]
AC70022A:  70.19( 54.86)  [H= 325, D= 38, S=100, I= 71, N= 463]   0.00 [N= 12]
AC70023A:  85.86( 62.15)  [H= 431, D= 15, S= 56, I=119, N= 502]   0.00 [N= 17]
AC70101A:  94.40( 86.64)  [H= 438, D=  7, S= 19, I= 36, N= 464]  10.00 [N= 20]
AC70102A:  90.81( 72.98)  [H= 494, D=  8, S= 42, I= 97, N= 544]  11.11 [N= 18]
AC70103A:  72.13( 62.37)  [H= 207, D= 21, S= 59, I= 28, N= 287]   0.00 [N= 10]
AC70201A:  89.73( 79.26)  [H= 463, D= 13, S= 40, I= 54, N= 516]   6.67 [N= 15]
AC70202A:  78.14( 66.95)  [H= 454, D= 58, S= 69, I= 65, N= 581]   0.00 [N= 11]
AC70203A:  85.85( 64.52)  [H= 467, D= 10, S= 67, I=116, N= 544]   0.00 [N= 18]
AC70301A:  86.68( 71.83)  [H= 397, D=  9, S= 52, I= 68, N= 458]   0.00 [N= 11]
AC70303A:  94.54( 86.55)  [H= 450, D=  7, S= 19, I= 38, N= 476]  26.32 [N= 19]
AC70304A:  66.54( 30.86)  [H= 179, D= 13, S= 77, I= 96, N= 269]   0.00 [N=  8]
CC70103A:  86.93( 74.40)  [H= 326, D=  9, S= 40, I= 47, N= 375]  11.11 [N=  9]
CC70109A:  85.35( 64.01)  [H= 332, D= 18, S= 39, I= 83, N= 389]   0.00 [N= 13]
CC70201A:  74.49( 60.58)  [H= 257, D= 40, S= 48, I= 48, N= 345]   0.00 [N=  7]
CC70212A:  75.93( 55.04)  [H= 407, D= 29, S=100, I=112, N= 536]   5.88 [N= 17]
CC70307A:  82.90( 68.91)  [H= 480, D= 38, S= 61, I= 81, N= 579]   9.09 [N= 11]
CC71001B:  77.74( 65.64)  [H= 681, D= 73, S=122, I=106, N= 876]   0.00 [N= 12]
CC71007A:  90.05( 78.23)  [H= 579, D= 12, S= 52, I= 76, N= 643]  42.31 [N= 26]
CC71008A:  80.88( 65.34)  [H= 567, D= 45, S= 89, I=109, N= 701]  11.76 [N= 17]
CC71016A:  87.95( 64.35)  [H= 518, D=  7, S= 64, I=139, N= 589]   0.00 [N= 16]
CC71035A:  89.84( 72.14)  [H= 345, D= 17, S= 22, I= 68, N= 384]   0.00 [N= 11]
CS70004B:  78.26( 68.04)  [H= 857, D= 83, S=155, I=112, N=1095]   0.00 [N= 14]
CS70010A:  77.31( 65.84)  [H= 310, D= 33, S= 58, I= 46, N= 401]   0.00 [N= 10]
CS70013A:  92.14( 80.34)  [H= 375, D=  8, S= 24, I= 48, N= 407]   0.00 [N=  7]
CS70020A:  83.11( 58.33)  [H= 369, D= 16, S= 59, I=110, N= 444]   0.00 [N= 11]
CS70023A:  89.86( 71.65)  [H= 824, D= 24, S= 69, I=167, N= 917]   0.00 [N= 14]
CS70025A:  82.51( 75.81)  [H= 382, D= 12, S= 69, I= 31, N= 463]   0.00 [N=  9]
CS70028A:  91.23( 76.16)  [H= 333, D=  4, S= 28, I= 55, N= 365]   7.69 [N= 13]
CS70034A:  82.43( 57.19)  [H= 516, D= 25, S= 85, I=158, N= 626]   5.88 [N= 17]
CS70047A:  88.35( 73.30)  [H= 364, D= 18, S= 30, I= 62, N= 412]   8.33 [N= 12]
CS70055A:  83.72( 72.97)  [H= 576, D= 46, S= 66, I= 74, N= 688]   0.00 [N= 11]
CS70059A:  87.36( 78.02)  [H= 318, D= 11, S= 35, I= 34, N= 364]  16.67 [N=  6]
CS70070A:  77.27( 60.06)  [H= 238, D= 17, S= 53, I= 53, N= 308]  12.50 [N=  8]
CS70074A:  92.09( 80.06)  [H= 291, D=  7, S= 18, I= 38, N= 316]   0.00 [N= 12]
CS70082A:  84.05( 69.94)  [H= 548, D= 23, S= 81, I= 92, N= 652]   0.00 [N=  8]
SC71005B:  86.84( 69.47)  [H= 475, D= 13, S= 59, I= 95, N= 547]   6.67 [N= 15]
SC71013A:  81.44( 65.11)  [H= 733, D= 44, S=123, I=147, N= 900]   0.00 [N= 14]
----------------------------- Overall Results -------------------------------
SENT: %Correct=6.22 [H=34, S=513, N=547]
PHONE: %Corr=83.48, Acc=67.99 [H=17832, D=958, S=2571, I=3309, N=21361]
=============================================================================
```

Figure 19: results of testing context dependent, 5 mixture HMMs

```
========================== HTK Results Analysis ============================
  Date: Thu Mar 27 15:27:50 1997
  Ref : /tmp/atrcdhresults.script.mlf.4270.2
  Rec : /tmp/atrcdhresults.script.mlf.4270
------------------------------ Speaker Results -----------------------------
spkr: %Corr( %Acc )  [ Hits, Dels, Subs, Ins, #Words] %S.Corr [ #Sent ]
----------------------------------------------------------------------------
AC70015A:  77.94( 64.88)  [H= 364, D= 23, S= 80, I= 61, N= 467]   0.00 [N= 14]
AC70016A:  69.44( 64.72)  [H= 250, D= 40, S= 70, I= 17, N= 360]   0.00 [N= 12]
AC70017A:  79.07( 75.97)  [H= 204, D= 10, S= 44, I=  8, N= 258]   9.09 [N= 11]
AC70019A:  84.20( 75.31)  [H= 341, D= 18, S= 46, I= 36, N= 405]   0.00 [N= 15]
AC70021A:  71.24( 57.08)  [H= 317, D= 24, S=104, I= 63, N= 445]   6.25 [N= 16]
AC70022A:  66.48( 59.58)  [H= 347, D= 91, S= 84, I= 36, N= 522]   7.69 [N= 13]
AC70023A:  83.07( 65.34)  [H= 417, D= 22, S= 63, I= 89, N= 502]   5.88 [N= 17]
AC70101A:  92.46( 86.85)  [H= 429, D=  9, S= 26, I= 26, N= 464]  15.00 [N= 20]
AC70102A:  88.60( 76.10)  [H= 482, D= 12, S= 50, I= 68, N= 544]  11.11 [N= 18]
AC70103A:  74.56( 68.29)  [H= 214, D= 25, S= 48, I= 18, N= 287]   0.00 [N= 10]
AC70201A:  88.37( 80.04)  [H= 456, D= 14, S= 46, I= 43, N= 516]   0.00 [N= 15]
AC70202A:  74.53( 69.88)  [H= 433, D= 69, S= 79, I= 27, N= 581]   0.00 [N= 11]
AC70203A:  86.03( 70.04)  [H= 468, D= 14, S= 62, I= 87, N= 544]   5.56 [N= 18]
AC70301A:  84.93( 71.62)  [H= 389, D= 18, S= 51, I= 61, N= 458]   0.00 [N= 11]
AC70303A:  92.23( 86.13)  [H= 439, D= 14, S= 23, I= 29, N= 476]  10.53 [N= 19]
AC70304A:  69.52( 42.01)  [H= 187, D= 14, S= 68, I= 74, N= 269]   0.00 [N=  8]
CC70103A:  84.27( 74.67)  [H= 316, D= 16, S= 43, I= 36, N= 375]   0.00 [N=  9]
CC70109A:  81.49( 72.49)  [H= 317, D= 26, S= 46, I= 35, N= 389]   7.69 [N= 13]
CC70201A:  74.49( 66.96)  [H= 257, D= 46, S= 42, I= 26, N= 345]   0.00 [N=  7]
CC70212A:  73.97( 56.62)  [H= 469, D= 36, S=129, I=110, N= 634]  11.11 [N= 18]
CC70307A:  82.04( 74.44)  [H= 475, D= 34, S= 70, I= 44, N= 579]  18.18 [N= 11]
CC71001B:  76.83( 68.15)  [H= 673, D= 76, S=127, I= 76, N= 876]   0.00 [N= 12]
CC71007A:  90.67( 81.18)  [H= 583, D= 14, S= 46, I= 61, N= 643]  38.46 [N= 26]
CC71008A:  80.46( 72.61)  [H= 564, D= 53, S= 84, I= 55, N= 701]  11.76 [N= 17]
CC71016A:  91.17( 76.23)  [H= 537, D=  6, S= 46, I= 88, N= 589]   6.25 [N= 16]
CC71035A:  85.68( 75.00)  [H= 329, D= 18, S= 37, I= 41, N= 384]  27.27 [N= 11]
CS70004B:  70.82( 65.75)  [H= 796, D=123, S=205, I= 57, N=1124]   0.00 [N= 15]
CS70010A:  75.31( 70.32)  [H= 302, D= 48, S= 51, I= 20, N= 401]   0.00 [N= 10]
CS70013A:  88.94( 80.59)  [H= 362, D= 16, S= 29, I= 34, N= 407]   0.00 [N=  7]
CS70020A:  84.23( 64.19)  [H= 374, D= 15, S= 55, I= 89, N= 444]   9.09 [N= 11]
CS70023A:  89.75( 77.97)  [H= 823, D= 30, S= 64, I=108, N= 917]   0.00 [N= 14]
CS70025A:  82.29( 76.67)  [H= 381, D= 28, S= 54, I= 26, N= 463]   0.00 [N=  9]
CS70028A:  88.22( 78.36)  [H= 322, D=  8, S= 35, I= 36, N= 365]   7.69 [N= 13]
CS70034A:  81.52( 67.55)  [H= 525, D= 31, S= 88, I= 90, N= 644]  11.11 [N= 18]
CS70047A:  86.41( 76.46)  [H= 356, D= 25, S= 31, I= 41, N= 412]   0.00 [N= 12]
CS70055A:  76.89( 71.51)  [H= 529, D= 86, S= 73, I= 37, N= 688]   0.00 [N= 11]
CS70059A:  86.81( 78.57)  [H= 316, D= 16, S= 32, I= 30, N= 364]  16.67 [N=  6]
CS70070A:  78.57( 66.23)  [H= 242, D= 26, S= 40, I= 38, N= 308]   0.00 [N=  8]
CS70074A:  85.13( 76.90)  [H= 269, D= 18, S= 29, I= 26, N= 316]  16.67 [N= 12]
CS70082A:  83.13( 72.09)  [H= 542, D= 30, S= 80, I= 72, N= 652]   0.00 [N=  8]
SC71005B:  83.91( 73.49)  [H= 459, D= 14, S= 74, I= 57, N= 547]   6.67 [N= 15]
SC71013A:  82.78( 69.78)  [H= 745, D= 50, S=105, I=117, N= 900]   0.00 [N= 14]
------------------------------ Overall Results -----------------------------
SENT: %Correct=7.44 [H=41, S=510, N=551]
PHONE: %Corr=81.61, Acc=71.44 [H=17600, D=1306, S=2659, I=2193, N=21565]
============================================================================
```

Figure 20: results of testing context dependent, 5 mixture HMMs using ATR SPREC recognizer

```
========================== HTK Results Analysis ==============================
  Date: Wed Apr  2 18:23:42 1997
  Ref : lists/test.monolabels.mlf
  Rec : lists/test.monoreclabels.mlf
------------------------------ Speaker Results -------------------------------
spkr: %Corr( %Acc )  [ Hits, Dels, Subs, Ins, #Words] %S.Corr [ #Sent ]
------------------------------------------------------------------------------
AC70015A:  69.59( 54.82)  [H= 325, D= 41, S=101, I= 69, N= 467]   0.00 [N= 14]
AC70016A:  57.78( 51.11)  [H= 208, D= 67, S= 85, I= 24, N= 360]   0.00 [N= 12]
AC70017A:  74.03( 67.44)  [H= 191, D= 28, S= 39, I= 17, N= 258]   0.00 [N= 11]
AC70019A:  62.96( 56.05)  [H= 255, D= 51, S= 99, I= 28, N= 405]   6.67 [N= 15]
AC70021A:  69.66( 61.57)  [H= 310, D= 53, S= 82, I= 36, N= 445]   0.00 [N= 16]
AC70022A:  59.83( 49.68)  [H= 277, D= 83, S=103, I= 47, N= 463]   0.00 [N= 12]
AC70023A:  69.12( 50.00)  [H= 347, D= 44, S=111, I= 96, N= 502]   0.00 [N= 17]
AC70101A:  77.37( 72.84)  [H= 359, D= 55, S= 50, I= 21, N= 464]   0.00 [N= 20]
AC70102A:  80.33( 67.46)  [H= 437, D= 33, S= 74, I= 70, N= 544]   0.00 [N= 18]
AC70103A:  58.19( 49.83)  [H= 167, D= 50, S= 70, I= 24, N= 287]   0.00 [N= 10]
AC70201A:  77.91( 71.51)  [H= 402, D= 53, S= 61, I= 33, N= 516]   6.67 [N= 15]
AC70202A:  65.40( 59.04)  [H= 380, D=113, S= 88, I= 37, N= 581]   0.00 [N= 11]
AC70203A:  73.16( 58.27)  [H= 398, D= 51, S= 95, I= 81, N= 544]   0.00 [N= 18]
AC70301A:  73.36( 63.54)  [H= 336, D= 50, S= 72, I= 45, N= 458]   0.00 [N= 11]
AC70303A:  80.25( 72.27)  [H= 382, D= 52, S= 42, I= 38, N= 476]  15.79 [N= 19]
AC70304A:  57.25( 33.83)  [H= 154, D= 26, S= 89, I= 63, N= 269]   0.00 [N=  8]
CC70103A:  71.47( 66.13)  [H= 268, D= 50, S= 57, I= 20, N= 375]   0.00 [N=  9]
CC70109A:  69.92( 60.41)  [H= 272, D= 49, S= 68, I= 37, N= 389]   0.00 [N= 13]
CC70201A:  65.80( 57.39)  [H= 227, D= 67, S= 51, I= 29, N= 345]   0.00 [N=  7]
CC70212A:  62.31( 51.87)  [H= 334, D= 75, S=127, I= 56, N= 536]   5.88 [N= 17]
CC70307A:  71.85( 62.52)  [H= 416, D= 68, S= 95, I= 54, N= 579]   0.00 [N= 11]
CC71001B:  65.98( 55.59)  [H= 578, D=143, S=155, I= 91, N= 876]   0.00 [N= 12]
CC71007A:  77.60( 69.83)  [H= 499, D= 60, S= 84, I= 50, N= 643]   7.69 [N= 26]
CC71008A:  66.62( 59.06)  [H= 467, D=117, S=117, I= 53, N= 701]   5.88 [N= 17]
CC71016A:  79.63( 66.55)  [H= 469, D= 41, S= 79, I= 77, N= 589]   0.00 [N= 16]
CC71035A:  74.74( 60.94)  [H= 287, D= 48, S= 49, I= 53, N= 384]   0.00 [N= 11]
CS70004B:  68.22( 60.91)  [H= 747, D=169, S=179, I= 80, N=1095]   0.00 [N= 14]
CS70010A:  62.84( 56.86)  [H= 252, D= 78, S= 71, I= 24, N= 401]   0.00 [N= 10]
CS70013A:  78.87( 70.52)  [H= 321, D= 47, S= 39, I= 34, N= 407]   0.00 [N=  7]
CS70020A:  72.30( 57.88)  [H= 321, D= 45, S= 78, I= 64, N= 444]   0.00 [N= 11]
CS70023A:  75.90( 65.10)  [H= 696, D= 92, S=129, I= 99, N= 917]   0.00 [N= 14]
CS70025A:  72.14( 68.68)  [H= 334, D= 41, S= 88, I= 16, N= 463]   0.00 [N=  9]
CS70028A:  81.64( 71.78)  [H= 298, D= 28, S= 39, I= 36, N= 365]   7.69 [N= 13]
CS70034A:  68.53( 56.87)  [H= 429, D= 78, S=119, I= 73, N= 626]   0.00 [N= 17]
CS70047A:  75.00( 63.35)  [H= 309, D= 48, S= 55, I= 48, N= 412]   8.33 [N= 12]
CS70055A:  69.19( 60.61)  [H= 476, D=114, S= 98, I= 59, N= 688]   0.00 [N= 11]
CS70059A:  74.73( 67.58)  [H= 272, D= 48, S= 44, I= 26, N= 364]   0.00 [N=  6]
CS70070A:  67.86( 58.77)  [H= 209, D= 43, S= 56, I= 28, N= 308]   0.00 [N=  8]
CS70074A:  77.53( 69.62)  [H= 245, D= 34, S= 37, I= 25, N= 316]   8.33 [N= 12]
CS70082A:  71.93( 62.88)  [H= 469, D= 82, S=101, I= 59, N= 652]   0.00 [N=  8]
SC71005B:  71.66( 61.24)  [H= 392, D= 59, S= 96, I= 57, N= 547]   6.67 [N= 15]
SC71013A:  67.22( 54.56)  [H= 605, D= 99, S=196, I=114, N= 900]   0.00 [N= 14]
------------------------------ Overall Results -------------------------------
SENT: %Correct=2.38 [H=13, S=534, N=547]
PHONE: %Corr=70.78, Acc=60.99 [H=15120, D=2673, S=3568, I=2091, N=21361]
==============================================================================
```

Figure 21: results of testing context independent, 15 mixture HMMs

```
========================== HTK Results Analysis ============================
  Date: Fri Apr 11 11:01:05 1997
  Ref : lists/test.monolabels.mlf
  Rec : lists/test.cdreclabels.mlf
------------------------------ Speaker Results -----------------------------
spkr: %Corr( %Acc )  [ Hits, Dels, Subs, Ins, #Words] %S.Corr [ #Sent ]
----------------------------------------------------------------------------
AC70015A:  65.95( 39.83)  [H= 308, D= 44, S=115, I=122, N= 467]   0.00 [N= 14]
AC70016A:  55.83( 36.94)  [H= 201, D= 60, S= 99, I= 68, N= 360]   0.00 [N= 12]
AC70017A:  67.05( 55.43)  [H= 173, D= 25, S= 60, I= 30, N= 258]   0.00 [N= 11]
AC70019A:  68.64( 52.10)  [H= 278, D= 40, S= 87, I= 67, N= 405]   0.00 [N= 15]
AC70021A:  64.49( 33.48)  [H= 287, D= 32, S=126, I=138, N= 445]   0.00 [N= 16]
AC70022A:  56.37( 36.72)  [H= 261, D= 68, S=134, I= 91, N= 463]   0.00 [N= 12]
AC70023A:  62.75( 37.65)  [H= 315, D= 50, S=137, I=126, N= 502]   0.00 [N= 17]
AC70101A:  76.08( 53.88)  [H= 353, D= 32, S= 79, I=103, N= 464]   0.00 [N= 20]
AC70102A:  76.10( 50.00)  [H= 414, D= 17, S=113, I=142, N= 544]   0.00 [N= 18]
AC70103A:  57.49( 39.72)  [H= 165, D= 47, S= 75, I= 51, N= 287]   0.00 [N= 10]
AC70201A:  76.55( 58.72)  [H= 395, D= 36, S= 85, I= 92, N= 516]   0.00 [N= 15]
AC70202A:  62.82( 50.60)  [H= 365, D= 92, S=124, I= 71, N= 581]   0.00 [N= 11]
AC70203A:  65.62( 40.99)  [H= 357, D= 41, S=146, I=134, N= 544]   0.00 [N= 18]
AC70301A:  69.21( 49.13)  [H= 317, D= 29, S=112, I= 92, N= 458]   0.00 [N= 11]
AC70303A:  75.00( 54.62)  [H= 357, D= 33, S= 86, I= 97, N= 476]   0.00 [N= 19]
AC70304A:  55.76( 23.42)  [H= 150, D= 29, S= 90, I= 87, N= 269]   0.00 [N=  8]
CC70103A:  64.53( 40.00)  [H= 242, D= 17, S=116, I= 92, N= 375]   0.00 [N=  9]
CC70109A:  62.98( 36.50)  [H= 245, D= 58, S= 86, I=103, N= 389]   0.00 [N= 13]
CC70201A:  57.97( 41.74)  [H= 200, D= 55, S= 90, I= 56, N= 345]   0.00 [N=  7]
CC70212A:  56.34( 30.41)  [H= 302, D= 65, S=169, I=139, N= 536]   0.00 [N= 17]
CC70307A:  66.84( 49.22)  [H= 387, D= 63, S=129, I=102, N= 579]   0.00 [N= 11]
CC71001B:  58.79( 40.64)  [H= 515, D=124, S=237, I=159, N= 876]   0.00 [N= 12]
CC71007A:  73.41( 44.63)  [H= 472, D= 42, S=129, I=185, N= 643]   0.00 [N= 26]
CC71008A:  56.78( 34.81)  [H= 398, D=101, S=202, I=154, N= 701]   0.00 [N= 17]
CC71016A:  71.31( 42.61)  [H= 420, D= 29, S=140, I=169, N= 589]   0.00 [N= 16]
CC71035A:  65.10( 48.70)  [H= 250, D= 47, S= 87, I= 63, N= 384]   0.00 [N= 11]
CS70004B:  54.89( 42.74)  [H= 601, D=171, S=323, I=133, N=1095]   0.00 [N= 14]
CS70010A:  64.09( 49.38)  [H= 257, D= 67, S= 77, I= 59, N= 401]   0.00 [N= 10]
CS70013A:  71.50( 48.16)  [H= 291, D= 32, S= 84, I= 95, N= 407]   0.00 [N=  7]
CS70020A:  68.92( 35.81)  [H= 306, D= 43, S= 95, I=147, N= 444]   0.00 [N= 11]
CS70023A:  66.63( 43.08)  [H= 611, D= 79, S=227, I=216, N= 917]   0.00 [N= 14]
CS70025A:  63.07( 47.52)  [H= 292, D= 42, S=129, I= 72, N= 463]   0.00 [N=  9]
CS70028A:  74.52( 49.04)  [H= 272, D= 17, S= 76, I= 93, N= 365]   0.00 [N= 13]
CS70034A:  57.19( 32.75)  [H= 358, D= 83, S=185, I=153, N= 626]   0.00 [N= 17]
CS70047A:  71.60( 48.79)  [H= 295, D= 36, S= 81, I= 94, N= 412]   0.00 [N= 12]
CS70055A:  65.70( 46.08)  [H= 452, D=101, S=135, I=135, N= 688]   0.00 [N= 11]
CS70059A:  69.23( 50.27)  [H= 252, D= 36, S= 76, I= 69, N= 364]   0.00 [N=  6]
CS70070A:  58.77( 37.01)  [H= 181, D= 46, S= 81, I= 67, N= 308]   0.00 [N=  8]
CS70074A:  75.32( 52.53)  [H= 238, D= 25, S= 53, I= 72, N= 316]   0.00 [N= 12]
CS70082A:  65.95( 46.47)  [H= 430, D= 81, S=141, I=127, N= 652]   0.00 [N=  8]
SC71005B:  63.44( 38.94)  [H= 347, D= 55, S=145, I=134, N= 547]   0.00 [N= 15]
SC71013A:  66.22( 48.78)  [H= 596, D=102, S=202, I=157, N= 900]   0.00 [N= 14]
------------------------------ Overall Results -----------------------------
SENT: %Correct=0.00 [H=0, S=547, N=547]
PHONE: %Corr=65.10, Acc=43.77 [H=13906, D=2292, S=5163, I=4556, N=21361]
============================================================================
```

Figure 22: results of testing context dependent, 5 mixture HMMs. Waveform files were parametrized by ATR SPREC software and converted into HTK format for use during training and testing.

References

[1] S.J. Young, J. Jansen, J.J. Odell, D. Ollason, and P.C. Woodland, "*The HTK book*", HTK v2.0 release, 1996.

[2] H. Singer, M. Tonomura, Q. Huo, J. Ishii, T. Fukada, M. Schuster, "*Baseline Acoustic Models for the Spoken Language Database (SDB/SLDB)*", TR-IT-0206, Interpreting Telecommunications Laboratory, ATR, Kyoto, Japan, March 1997.

[3] S.J. Young, P.C. Woodland, and W.J. Byrne, "*HTK: Hidden Markov Model Toolkit V1.5 – Reference Manual*", HTK v1.5 release, 1993.

[4] S.J.Young, J.J.Odell, and P.C.Woodland, "*Tree-Based State Tying for High Accuracy Acoustic Modelling*", Proc. DARPA workshop, pp.286-291, 1994.

[5] T. Hori, M. Katoh, A. Ito, and M. Kohda, "*A Study on HM-Net using Successive State Splitting based on Phonetic Decision Tree*", Technical Report SP96-22, IEICE, June 1996.

[6] G. Anderson and P. Anderson, "*The UNIX C Shell Field Guide*", Prentice Hall, 1986

[7] Shigeki Sagayama, "*Speech Analysis and Phoneme Label Conversion for Speech Recognition Research*", TR-I-0347, Interpreting Telecommunications Laboratory, ATR, Kyoto, Japan, March 1993.