INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. • Suite 600 • Berkeley, California 94704-1198 • (510) 643-9153 • FAX (510) 643-7684

The Voice Mail Digits and Their Performance on ICSI's Hybrid HMM/ANN System

Rainer Klisch*

TR-96-013

April 1996

Abstract

This report describes how we used ICSI's Hidden Markov Model (HMM) / Artificial Neural Network (ANN) speech recognition system to evaluate the Voice Mail (VM) digits corpus. We will present the new database, discuss the structure of the HMM/ANN recognizer, and finally report on the recognition performance we achieved in this initial work.

*Institute for Communication Systems and Data Processing (IND) at RWTH Aachen, Germany

Acknowledgments

I would like to thank my advisor Morgan, who made my stay at ICSI possible, my officemate Su-Lin Wu for all her help and licorice, Eric Fosler, the wizard who cares about every question and knows all the answers, David Johnson for his in- and outdoor advice, Brian E. Kingsbury for listening to my whining and for his helpful ideas, and all the other members of the realization group at ICSI for providing an easy and inspiring working atmosphere.

We also thank Siemens, in particular Joachim Köhler, for providing the VM digits database.

This research is supported by a National Science Foundation grant MIP-9311980 to the International Computer Science Institute.

Contents

1 Introduction

At ICSI there is a rich history of speech recognition research aimed at tasks with medium to large size vocabularies like the Resource Management database (about 1000 words) and the Wall Street Journal database (> 5000 words). Along with these tasks evolved the structure of ICSI's hybrid speech recognition system based on Hidden Markov Models (HMM) and Artificial Neural Networks (ANN). In recent work [6] this framework was applied to databases with small vocabularies like the Bellcore Isolated Digits (13 words), the TI Connected Digits (11 words), and Numbers'93(95) (36 resp. 100 words). This report describes how we use this recognition scheme for another small vocabulary corpus, the Voice Mail (VM) Digits developed by Siemens (Munich). As this database has an identical vocabulary to the Bellcore Digits, the main goal was to find out how the HMM/ANN recognizer would perform on inter-database experiments, i.e. training on the Bellcore Digits and testing on the VM Digits. This text will give a short overview of the digits databases with regard to size, speaker characteristics, and noise. Furthermore I will describe the front-end processing, ANN structure, the grammar and lexicon, and the software setup which was used for the experiments with the digits databases in this work.

2 The Isolated Digits Databases

The vocabulary of the digits databases comprises thirteen words: The digits "zero", "one", "two", ..., "nine", and "oh","no" and "yes". A total of about 2600 utterances, each consisting of one isolated word, was collected over the public telephone network in the USA for each database. Roughly 200 native adult speakers of both genders contributed to the speech data. The speech has been sampled at 8kHz and is supplied as 16-bit wave files.

2.1 The Bellcore Isolated Digits

This database was composed at Bellcore Labs. These digits are spoken by very disciplined and cooperative speakers. There is generally very low background noise in the speakers' environments. Common sources of noise are low quality communication channels and occasional bursts, probably due to overmodulation of the microphone. This database has been evaluated on ICSI's speech recognition system in the scope of earlier work. In the experiments reported in [6] this data was split into four jackknife-cuts that were used for training and testing the HMM/ANN speech recognition system. In those experiments starting or ending silence surounding every utterance were excised by hand before the data was fed into the recognition system. In the work reported here we clustered the four jackknife-cuts into one big training set. For consistency, some tend to call it "historical reasons", we again chose to cut off surrounding noise prior to further processing. This leaves us with about half an hour of acoustic input from this database.

2.2 The Voice Mail Isolated Digits

This database was supplied by Siemens/Munich. The 87 male and 119 female native speakers who contributed to this database were situated in their "natural" environment. This

makes this database a very realistic one, but also implies substantial noise. Apart from additional sounds - e.g. slamming doors, clicks, keyboard noise, beeps - channel conditions as well as speaking effort differed over a wide range. Therefore amplitude gain and SNR may considerably vary for the utterances of two distinct speakers. Moreover, the speech recognition system will have to cope with additional "garbage" sounds. To illustrate these phenomena the wave forms of three speech samples have been included which show the distortions intoduced by keyboard noise (fig. 1), coughing (fig. 2) and beeps (fig. 3).

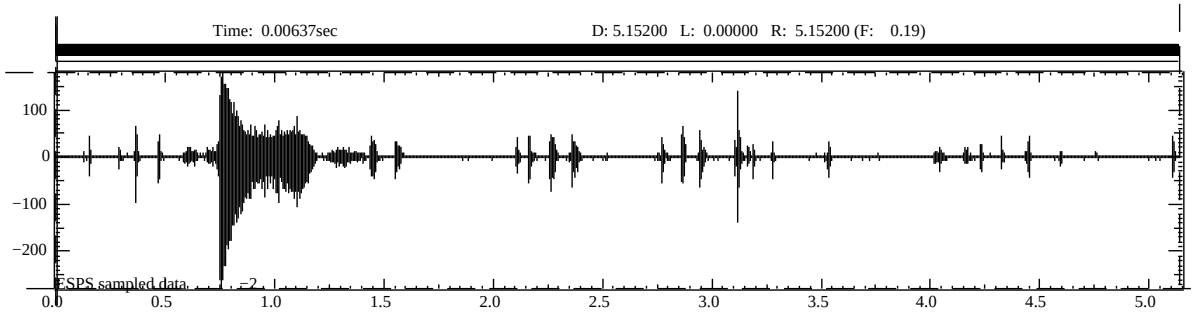

Figure 1: "five" + keyboard noise

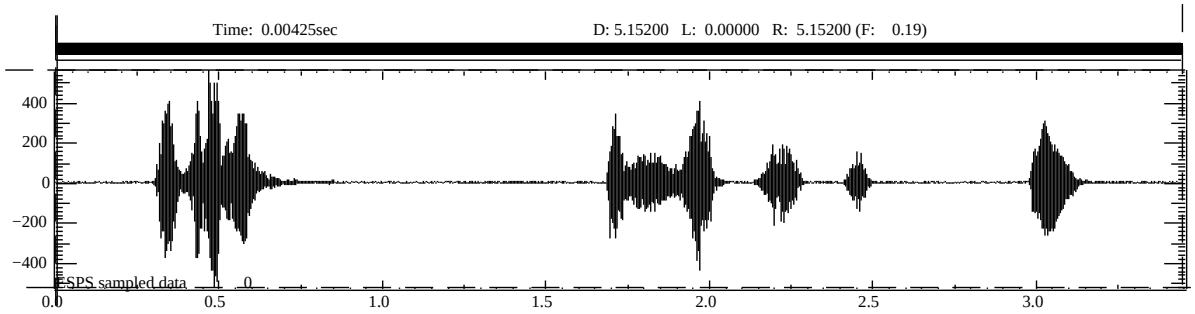

Figure 2: "oh" + coughing

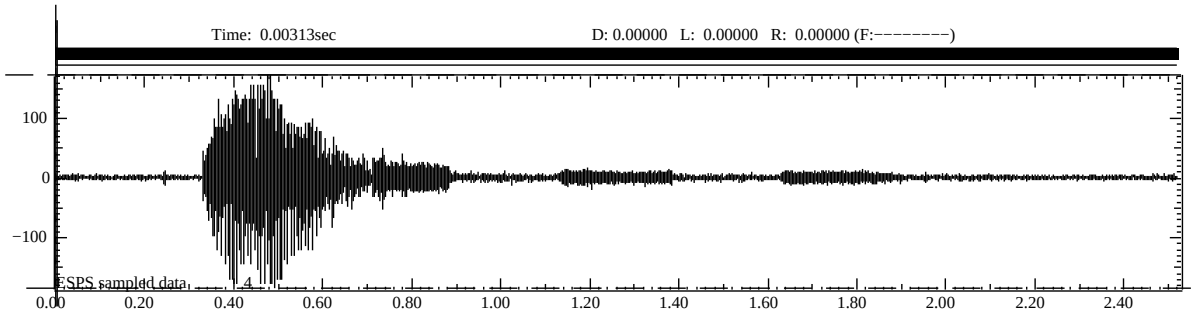

Figure 3: "oh" + beeps

Twelve of the 2597 utterances were rejected as they contain multiple words. The remaining 2585 speech samples amount to roughly one hour of acoustic information half of that falls into the category of "noisy silence", i.e. starting and ending stretches of non-speech which surround the articulation of the digit. In the experiments described later in this report we use this data without any modification – we do not delete any stretches of silence like with the Bellcore Digits – to test the speech recogniton system.

3 The HMM/ANN Speech Recognition System

The speech recognition system developed at ICSI is a hybrid structure joining a Hidden Markov Model (HMM) and an Artificial Neural Network to model speech. The system architecture used in this experiments is presented in fig. 4.

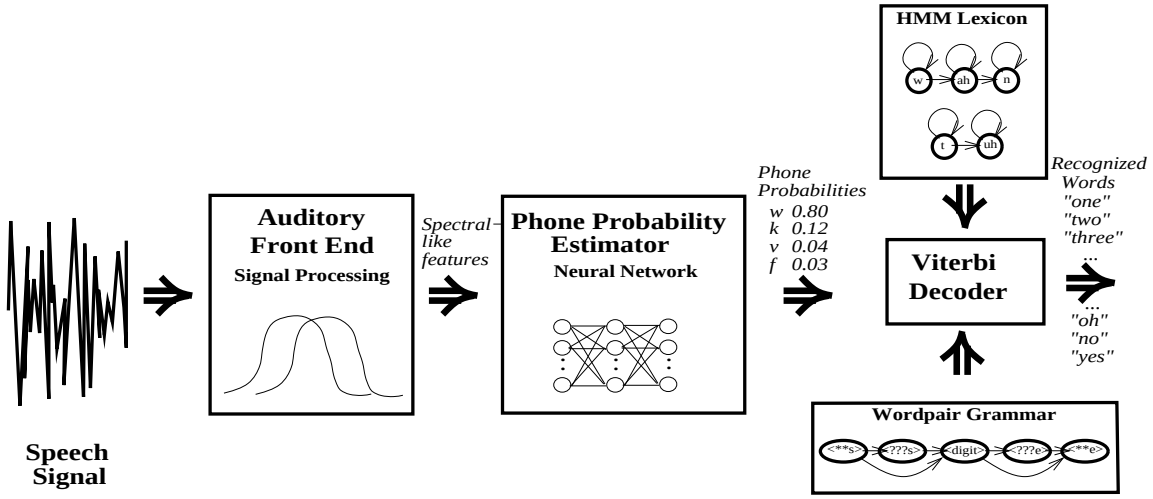

Figure 4: Hybrid HMM/ANN Speech Recognition System

The sampled speech data is input to the front-end processing stage. At this point noise reduction and feature extraction is performed. These features are then piped into the artificial neural network to compute the posterior probabilities for each phone (of a specific phone-set). On the basis of the HMM lexicon, the wordpair grammar and the phone probabilities, the recognizer then determines the most probable word.

3.1 Front-End Processing

For the digits experiments we normally group the speech data into frames of 25ms width with 12.5ms overlap. By means of Perceptual Linear Prediction (PLP) we extract acoustic/auditory features within every frame. PLP is an extension of linear predictive analysis that takes into account some aspects of human sound perception. In this work 8 PLP and 8 delta PLP coefficients as well as delta energy are computed for the feature vector of each frame. To cope with "convolutional noise" at the front end, i.e. linear distortion to the communication channel for the speech input, we use RASTA (see [3]) to filter this noise in the logarithmic power spectrum domain. In the case that speech is also significantly corrupted by additive noise we apply Jah-RASTA which adapts to the actual prevailing type of noise: it performs a more logarithmic-like transform for prevailing convolutional noise and linear-like processing for severe additive noise. The nonlinear transform used by JRASTA is a function of the noise level. In our experiments we normally train the recognition system with clean speech assuming the same low noise level for all utterances, i.e. we use the same nonlinear transform for all utterances in the training set. During the evaluation of the test data we adapt the nonlinear transform in compliance with the actual noise level and use

spectral mapping to compensate for this source of variability. The mapping coefficients for this process are determined from a subset of the training data. The "standard" mapping coefficients used for testing with the Bellcore Digits performed poorly for the VM Digits. A visual analysis (using command line C.4 of the appendix) of the resulting front-end output revealed severely "smeared" spectra. After generating improved mapping coefficients with the command C.2 we achieved the anticipated performance. For detailed information on spectral mapping in the scope of JRASTA, see [5].

3.2 The ANN - A Multiple Layer Perceptron

The Artificial Neural Network we use for computing the posteriori probability for the phones (given a sequence of acoustic features) is a multiple layer perceptron (MLP), shown in fig. 5. It comprises one hidden layer of 200 units that are fully connected to the 153 units of the input layer. The input to the MLP consists of the feature vectors for the actual frame and eight surrounding frames. This introduces 60ms of acoustic context on both "sides" of the actual frame at the MLP input. The 61 units of the output layer represent the 61 phones of the TIMIT61 phoneset (see appendix A) which is applied in this work.

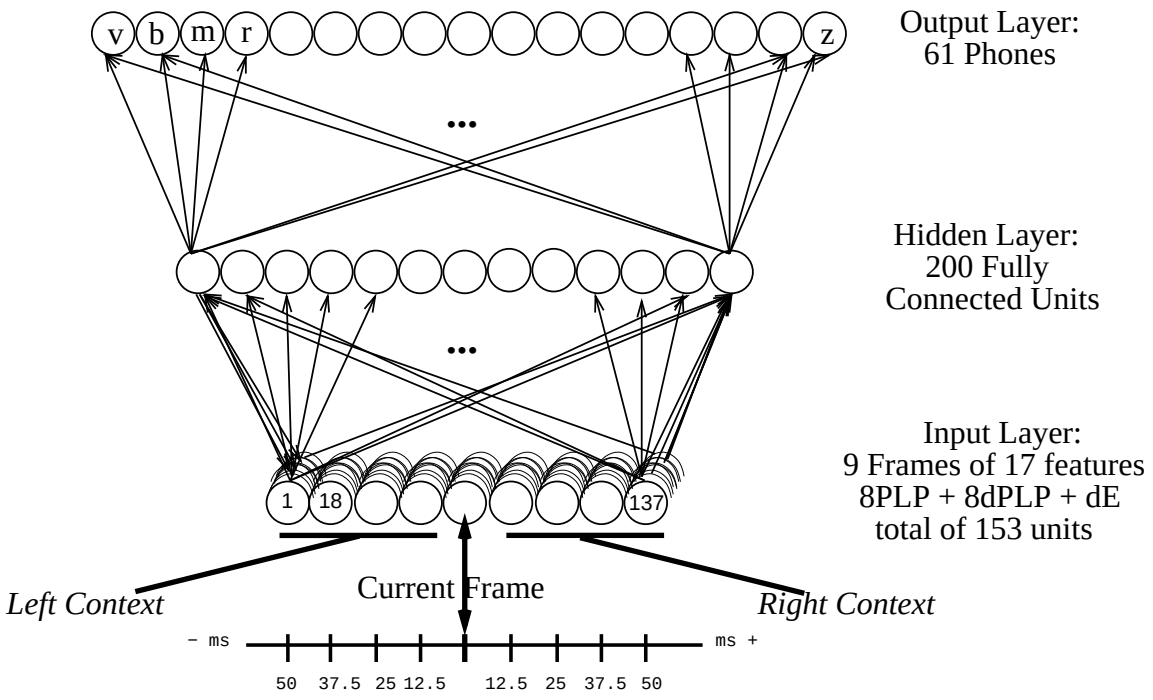

Figure 5: Multiple Layer Perceptron

3.3 The Recognizer

The recognizer Y0 ("why not") reads the phone probability vectors generated by the MLP to determine the "correct" (sequence of) word(s). To find the most probable path through

a probablelistically scored time/state lattice, Y0 applies dynamic programming in form of the Viterbi algorithm. The phonological and syntactical information from the HMM lexicon and the wordpair grammar provide a constraining framework to assist the recognizer with this task.

3.4 Hidden Markov Model Lexicon and Wordpair Grammar

In the Hidden Markov Model (HMM) lexicon we model the phone states and transistions between these states for every word. A typical example for a single pronunciation HMM lexicon is presented in fig. 6. Note that the minimum number of states in a specific phone for a given word is predefined by the lexicon whereas the maximum number of states is unrestricted, as selfloops allow an arbitrary long stay in any given state. We apply no sophisticated way to compute the transistion probabilities - they are generally set to 1/num_of_trans - as former experiments revealed no improvements for weighted transitions. The *garbage word* uses a virtual phone, the so called garbage phone. The probability of being in a *garbage state* is computed by averaging over the probabilities of the actual 2nd to (n+1)th best ranking phone. We achieved the best results for this task with $5 \leq n \leq 10$.

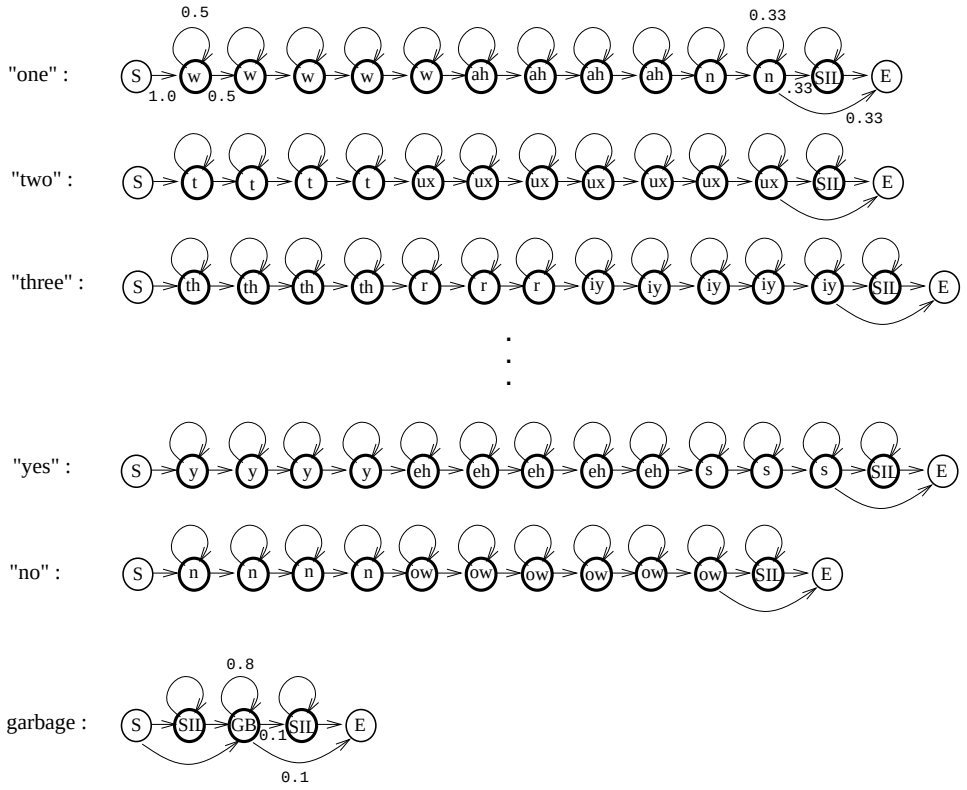

Figure 6: Markovian Single Pronunciation Lexicon

The introduction of the *garbage word* considerably increased recognition performance, as it allows the recognizer to account for additional sounds in the utterance that are not due to

the spoken digit. The corresponding extension of the wordpair grammar (fig. 7) establishes an optional *garbage state* before (`???s`) and after (`???e`) the proper digit utterance.

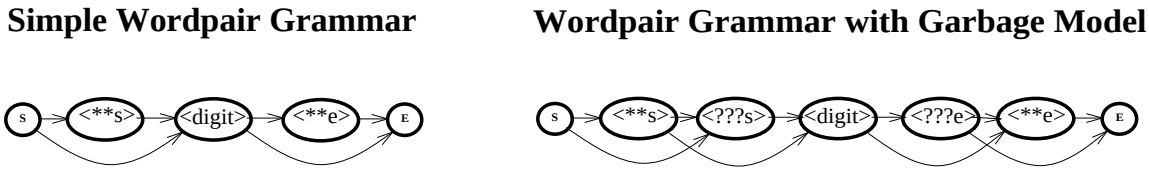

Figure 7: Extending the Wordpair Grammar

In an alternative approach we considered a *"Minimum-Maximum-Duration Lexicon"* to improve recognition in noisy environments. To generate this lexicon we determined the minimum and maximum length of every phone in every word from the training set, i.e. the Bellcore Digits. We expected that the additional constraint of a fixed maximum length for every digit would help the recognizer to classify which part of the utterance was due to the spoken digit and which originated from noise. Although we observed a significantly improved recognition rate over the case of the HMM-lexicon without the *garbage word*, the performance was still slightly inferior to the above mentioned HMM-lexicon with the *garbage word* and modified wordpair grammar.

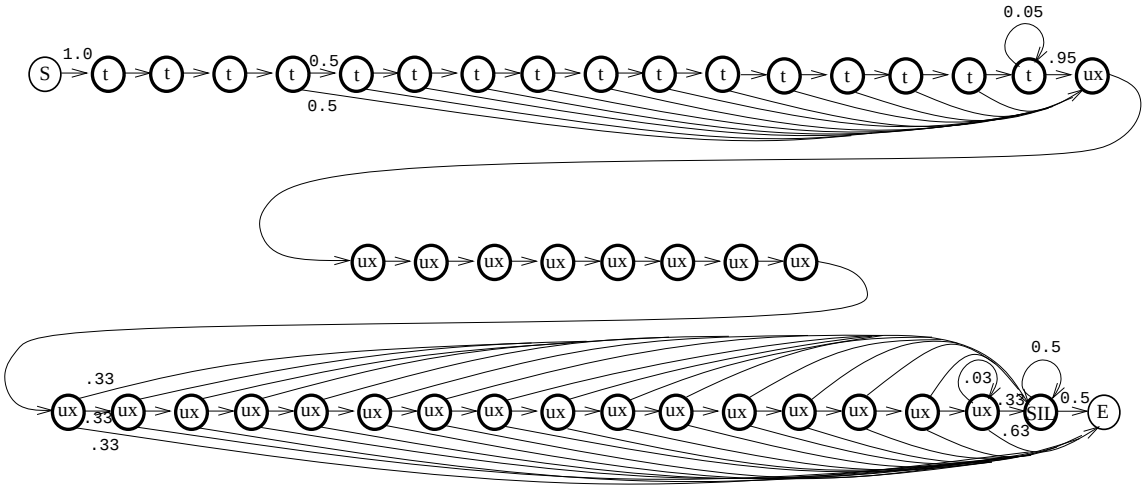

Figure 8: The Word "two" of the Minimum-Maximum-Duration Lexicon

4 Experiments

4.1 Software

The experiments were run under the control of the `isr_embed` script which initializes, runs and coordinates the ICSI Speech Recognition software for embedded training of an HMM/MLP recognition system. This script reads the individual parameter settings for the

experiment from the files `boot.mk` and `files.mk` (see appendix B.1 and B.2 for examples) and then starts multiple `pmake` jobs to execute all necessary tasks.

4.2 The Experiment Configuration

According to the discussion in the preceeding sections we focused on the following experiment outline.

Training Data: Bellcore Digits (2597 utterances)

Test Data: Voice Mail Digits (2585 utterances)

Front-End Processing: RASTA, JRASTA using 8 PLP + 8 ΔPLP+ΔEnergy features

MLP: 3 layers: 153 inputs, 200 completely connected hidden units, 61 outputs

Phoneset: Timit61

Lexicon: Single Pronunciation HMM Lexicon with garbage word

Grammar: Wordpair Grammar with optional starting/ending garbage state

Bootnet: Random initialization or a net bootstrapped from the handlabeled NTIMIT database applying RASTA/JRASTA

Bootlexicon: Hand-tuned lexicons taken from earlier experiments [6]

Number of Iterations: 7

The experimental results are reported in table 1 . We varied the front-end processing as well as the bootnet to learn about the performance behavior. For each experiment we state the iteration with the lowest error rate. From these results we can see that the performance for

Table 1: Recognition Results

Exp. #	Frontend	Bootnet	Iteration	Error Rate in %
1	RASTA	ntimit-rasta	1	2.2
2	RASTA	rand	1	3.2
3	JRASTA	ntimit-rasta	5	2.1
4	JRASTA	ntimit-jrasta	5	2.6
5	JRASTA	rand	3	2.6

RASTA and JRASTA is roughly the same for this special recognition task. To allow some further insight into the recognition errors we included the confusion matrix for experiment 3 on the next page.

Table 2: Confusion Matrix for Experiment 3

	one	two	three	four	five	six	seven	eight	nine	zero	oh	no	yes
one	197	.	.	.	1	1	.	.	1	.	.	.	.
two	.	197	.	.	.	.	.	.	.	.	.	1	.
three	.	2	193	.	.	.	.	3	.	.	1	.	.
four	1	.	.	193	.	.	.	.	.	.	.	1	.
five	1	.	.	1	194	.	.	1	3	.	.	.	.
six	.	1	1	.	.	191	3	4	.	.	.	.	.
seven	.	.	.	.	.	1	198	.	1	.	2	.	.
eight	.	1	1	.	.	2	.	194	.	.	.	.	.
nine	1	.	1	.	2	.	1	.	190	.	.	2	.
zero	.	1	.	.	.	.	.	.	.	199	.	.	.
oh	.	2	.	.	.	.	.	.	.	.	193	2	.
no	.	.	.	.	.	.	.	.	1	2	5	190	.
yes	.	.	.	.	.	.	.	1	.	1	.	.	199

5 Hardware Shift: From RAP to SPERT

At ICSI we use specialized hardware to achieve a speedup for neural network computations. In earlier approach, the RAP (Ring Array Processor), clusters off-the-shelf DSPs were used to improve computation performance. The latest design, the SPERT board, makes use of a custom-made, state-of-the-art, 16-bit, vector processor that has been developed at ICSI. When we switched from RAP to SPERT, we faced the situation that the weights of a net trained on the RAP in 32-bit floating point precision had to be loaded into the 16-bit fixed point architecture of the SPERT. The default setting on the SPERT would allow a number range of (-1.0 , 1.0), i.e. there are no bits reserved for the exponent. In this scenario, weights that lie outside this range will experience saturation onto loading to the SPERT. A histogram for the weights between input layer and hidden layer for a typical net, as shown in fig. 9, reveals that less than 2% of the weights would be affected. The question was whether it is more important to avoid saturation for even only a few weights or to achieve a more precise representation for a large amount of smaller numbers. To clarify this, we ran a sequence of identical experiments where we used a net trained on the RAP for testing on the SPERT and only changed the number of bits for the exponent thus controlling the number of weights that would saturate. The results are shown in Tabel 3 below.

Table 3: Experiment Sequence

Recognition	Substitutions	Error Rate in %
RAP = reference	63	2.4
SPERT, exp = 0 bit	90	3.5
SPERT, exp = 2 bit	64	2.5
SPERT, exp = 3 bit	63	2.4

From these results it is obvious that it is indeed more important to prevent even just a

relatively small number of weights from saturating than to achieve higher precision for the presentation of large number of weights.

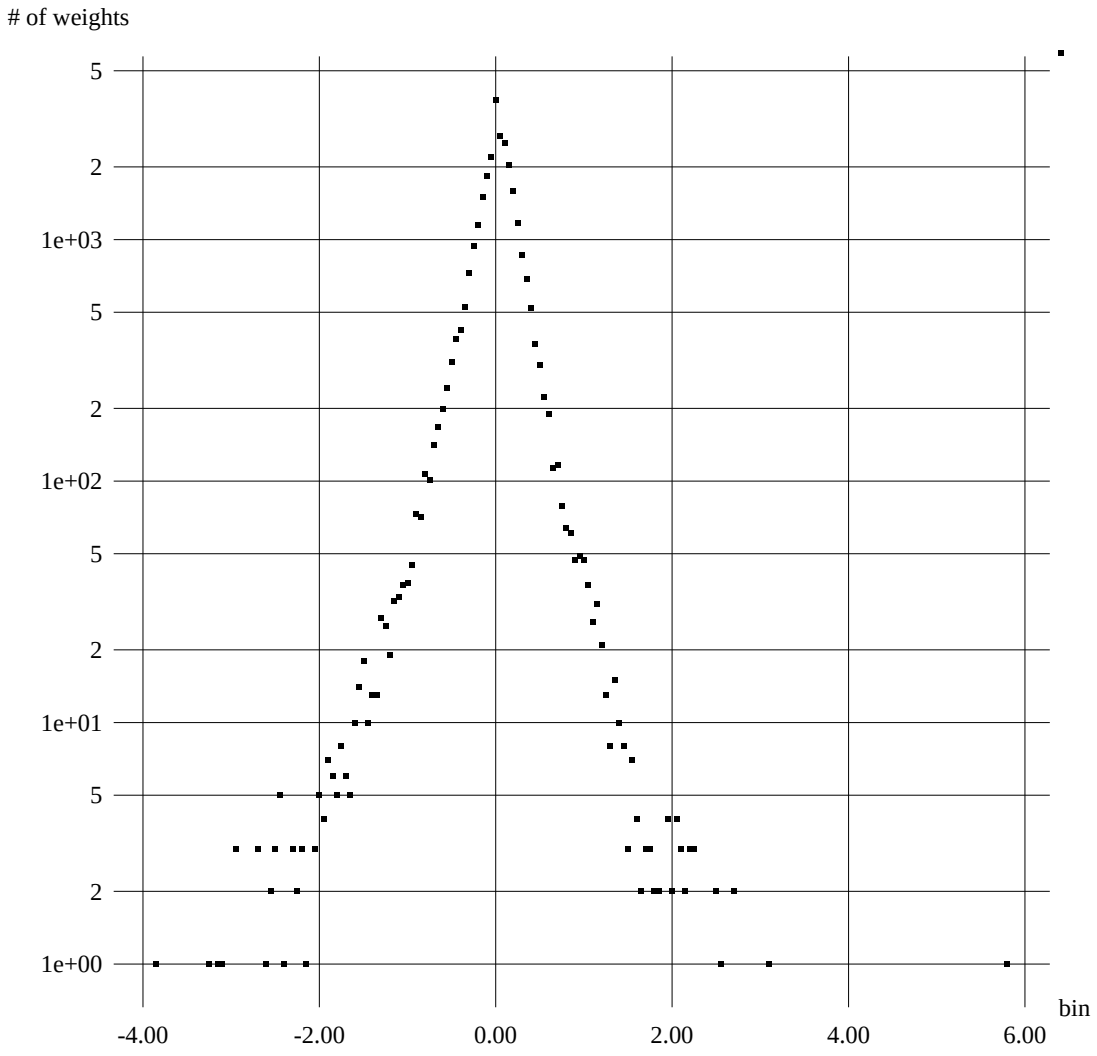

Figure 9: Histogram for weights between input and hidden layer

6 Conclusion

In this paper we have described the two digits databases from Bellcore and Siemens. We have pointed out that the latter is more realistic with regard to speaker behavior and background noise. The discussion of ICSI's hybrid HMM/ANN system and its application to the digits databases has revealed how we have successfully used a speech recognition system originally developped for large vocabulary size tasks for small corpora databases. The experiments of this work have been focused on inter-database tasks where we used the

Bellcore Digits to train the recognizer and subsequently tested on the VM Digits. We have presented the *garbage model* as an effective means to handle real-life-noise in the digits database. Comparing the performance of our speech recognition system on this task (2.1% word error rate) with the results reported in [6] for the experiments on the jackknife-cuts of the Bellcore Digits (1.2% word error rate) it is found, that the use of two distinct databases for training and testing is a more difficult task for the speech recognizer. It is hoped that the introduction of transition-based acoustic information and the implementation of context-dependent networks will further improve the recognition rate.

A TIMIT61 phonset

Num	Phone	Num	Phone	Num	Phone
0	b	20	th	40	ey
1	d	21	v	41	ae
2	g	22	dh	42	aa
3	p	23	m	43	aw
4	t	24	em	44	ay
e 5	k	25	n	45	ah
6	dx	26	nx	46	ao
7	bcl	27	ng	47	oy
8	dcl	28	eng	48	ow
9	gcl	29	en	49	uh
10	pcl	30	l	50	uw
11	tcl	31	el	51	ux
12	kcl	32	r	52	er
13	jh	33	w	53	axr
14	ch	34	y	54	ax
15	s	35	hh	55	ax-h
16	sh	36	hv	56	ix
17	z	37	iy	57	h#
18	zh	38	ih	58	pau
19	f	39	eh	59	epi
				60	q

B Experimental Set Up

B.1 Sample boot.mk

```
# boot.mk file for embedded training of VM database initializing from NTIMIT
old_pfile_stamp =
old_weights     = $(HOME)/VM/boot_ntimit/ntimit.r8-25ms-noe.200.w
old_priors      = /n/marmite/db/lingling/DIGIT/train1/Priors/jcleantrain1fv0.prior
old_lexicon     = /n/marmite/db/lingling/DIGIT/lexicon/LEX.digits.cddur.1.jfv0
```

B.2 Sample files.mk

```
#ifndef HAVE_FILES_MK
HAVE_FILES_MK  =

run = $(EXPERIMENT)

BOOGIE_EXPORTS=diskful 32meg !next !iris !mips !SunOS5.4 !SunOS5.3 !SunOS5.5
QN_FFP_EXPORTS=diskful 32meg !next !iris !mips !SunOS5.4 !SunOS5.3 !SunOS5.5
RECOG_EXPORTS=diskful 32meg !next !iris !mips !SunOS5.4 !SunOS5.3 !SunOS5.5
```

```
MLP_TRAIN_ARCH=spert
MLP_TRAIN_HOST=marmite
MLP_FFP_ARCH=spert
MLP_FFP_HOST=marmite
QN_FORWARD = qnforward-v0_11
QN_TRAIN = qntrain-v0_11
MAKE_SENT_RANGES = make_sent_ranges batchsize=500
MAXITER = iter7

rapdir  = ./spert
rundir  = ./run
pfile       = $(HOME)/DIGITS/data2600/digits2597.jrasta8+d.pfile
norms       = ./$(run).norms
sents       = $(HOME)/DIGITS/data2600/answer2597
init_weights= $(HOME)/VM/boot_ntimit/ntimit.r8-25ms-noe.200.w
test_pfile  = $(HOME)/VM/pfiles/vmdigits.cut1-4.jrasta8e+d_11j-map.pfile
test_sents  = $(HOME)/VM/list/answer.cut1-4
fpause      = $(HOME)/VM/lexicon/fpause.void
silence     = $(SPEECH_DIR)/data/lexicon/numbers.95.silence.icsi61
fun_words   = $(HOME)/VM/lexicon/fun_words.void
garbage_word= $(HOME)/VM/lexicon/garbage_s_e.timit61
phoneset    = $(SPEECH_DIR)/data/phonesets/timit61.phset

mlp_window_width  = 9
mlp_window_offset = -4
mlp_num_hidden    = 200
mlp_num_output    = 61
mlp_num_input     = 153
mlp_first_ftr     = 1
mlp_num_ftrs      = 17

fvit_wordtranspenalty  = 1.0
statedecode            = true
samplerate             = 8000
framestep              = 100
startoffset            = 4
endoffset              = 4
task                   = rm
fvit_maxactivewords    = 10
fvit_startpruningframe = -1
fvit_transcut          = 0
fvit_beamwidth         = 200
fvit_phoneprune        = 0
printlikelihoods       = false
```

```
y0_opt_fvit_params     = "verbose=true"
wordtranspenalty  = 1
maxactivewords    = 10
startpruningframe = -1
transcut          = 0
beamwidth         = 200
phoneprune        = 0
lmscale           = 1.0
train_cache_sents = 200

wpbegin = $(HOME)/VM/grammar/wp_begin_gab_s_e
wpcont  = $(HOME)/VM/grammar/wp_cont_gab_s_e
wpend   = $(HOME)/VM/grammar/wp_end_gab_s_e
garbage = 10
amscale = 1.0

# YOU DON'T NEED TO CHANGE ANYTHING AFTER THIS
#
# Directories where stuff is generated
bob_dir    ?= $(rundir)/net
pfile_dir  ?= $(rapdir)
lna_dir    ?= $(rapdir)
weight_dir ?= $(rundir)/net
prior_dir  ?= $(rundir)/net
word_dir   ?= $(rundir)/out
align_dir  ?= $(rundir)/out
label_dir  ?= $(rundir)/out
lex_dir    ?= $(rundir)/lex
pron_dir   ?= $(rundir)/lex
dur_dir    ?= $(rundir)/lex
result_dir ?= $(rundir)/out

# these are filled in by the top-level makefile
job        ?= foo
new_job    ?= new_$(job)
test_job   ?= test_$(job)

# Bootstrap files
old_pfile_stamp ?= $(pfile_dir)/$(old_job).pfile.stamp
old_weights     ?= $(weight_dir)/$(old_job).wts
old_priors      ?= $(prior_dir)/$(old_job).priors
old_lexicon     ?= $(lex_dir)/$(old_job).lex
old_lnafile ?= $(lna_dir)/$(old_job).lna

# Generated files
```

```
lnafile   ?= $(lna_dir)/$(job).lna
align     ?= $(align_dir)/$(job).align
labels    ?= $(label_dir)/$(job).label
wordpaths ?= $(word_dir)/$(job).word
prons     ?= $(pron_dir)/$(job).prons
pfile_stamp ?= $(pfile_dir)/$(job).pfile.stamp
priors      ?= $(prior_dir)/$(job).priors
durations   ?= $(dur_dir)/$(job).dur
weights     ?= $(weight_dir)/$(job).wts
lexicon     ?= $(lex_dir)/$(job).lex
fvit_lexicon = $(lexicon)
test_lnafile   ?= $(lna_dir)/$(EXPERIMENT).lna
test_results   ?= $(result_dir)/$(EXPERIMENT).result
test_scores    ?= $(result_dir)/$(EXPERIMENT).score

#endif HAVE_FILES_MK
```

C Command Lines

C.1 RASTA Processing

```
/u/drspeech/sun4/bin/rasta  -e -m 8 -n 9 -w 25.000000 -s 12.500000 -S 8000
-A -F -i <infile>  -o <outfile>
```

C.2 JRASTA Map File Generation

```
create_mapping fileList=mapping.list546 jahList="2.15e-4 1.0e-4 4.64e-5 2.15e-5
1.0e-5 4.64e-6 2.15e-6 1.0e-6 4.64e-7 2.15e-7 1.0e-7 4.64e-8 2.15e-8 1.0e-8
4.64e-9 2.15e-9 1.0e-9" cleanJah=1.0e-6 mapFile=map_17j.dat
rasta0pts="-m 8 -n 9 -w 25.000000 -s 12.500000 -S 8000 -F"
```

C.3 JRASTA Processing

```
/u/drspeech/sun4/bin/rasta (-M) -e -m 8 -n 9 -w 25.000000 -s 12.500000 -S 8000
-J -f ~rklisch/VM/mapping/map_17j.dat  -A -F -i <infile> -o <outfile>
```

C.4 "RASTA-Gramm"

```
cat <(j)rasta-file> | ~/bin/cep2spec.pl format=ascii order=8 nfilt=17 |
addfeahd -a -C /dev/null/ ~/bin/rasta.format - - | image -f"rasta" -B 0 - &
```

References

[1] Herve Boulard and Nelson Morgan, *"Connectionist Speech Recognition: A Hybrid Approach"*, The Kluwer International Series in Engineering and Computer Science, 1994

[2] Nelson Morgan and Herve Boulard, *"Continous Speech Recognition - An introduction to the hybrid HMM/connectionist approach"*, IEEE Signal Processing Magazin, May 1995.

[3] Hyneck Hermansky and Nelson Morgan, *"RASTA Processing of Speech"*, IEEE transactions on speech and audio processing, Vol. 2, No.4, Oct. 1994.

[4] Hyneck Hermansky, *"Perceptual Linear Predictive (PLP) Processing for Speech"*, J. Acoust. Soc. Am., pp. 1738-1752, 1990.

[5] Grace C.H. Tong, *"Combating Additive Noise and Spectral Distortion in Speech Recognition Systems with JAH-RASTA"*, Masters Thesis, University of California at Berkeley, 1994.

[6] Kristine W. Ma, *"Applying Large Vocabulary Hybrid HMM-MLP Methods to Telephone Recognition of Digits and Natural Numbers"*, Masters Thesis, University of California at Berkeley, 1995.

[7] Dan Jurafsky, *"The Beginners Guide to the ICSI Speech Software"*, July 1995.