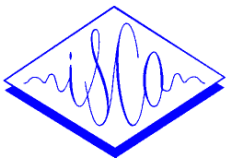

2nd International Conference on
Spoken Language Processing (ICSLP 92)
Banff, Alberta, Canada
October 12-16, 1992

HARK: AN EXPERIMENTAL SPEECH RECOGNITION SYSTEM

David M. Goblirsch and Toffee A. Albina
dmgob@mitre.org and talbina@mitre.org

The MITRE Corporation
McLean VA 22102

Abstract

As part of its internal research program, MITRE has developed an experimental speech recognition system for small-vocabulary simple-grammar applications. This system, called *Hark*, has three parts: a signal processor for extracting feature vectors from the acoustic signal, a neural network for classifying the feature vectors, and a dynamic programming module for decoding the utterance. *Hark* is speaker independent and handles continuous speech with pauses after sentences. This paper provides an overview of the *Hark* system.

10.21437/ICSLP.1992-191

1 Introduction

This paper describes an experimental speaker-independent continuous-speech (within sentences) recognition system that was developed as part of MITRE's internally funded research program. The system is called *Hark*, which means "to listen attentively."

Hark is an application-independent kernel around which application-specific speech recognition systems can be built. To develop a particular application, the user creates three ASCII files that describe the vocabulary and grammar for the task. The application program calls the *Hark* system each time that speech is to be processed. *Hark* reads the three application-specific files at run-time. The current version of *Hark* is push-to-talk.

Hark was developed on a Sun SPARC platform, but has also been ported to HP9000 series workstations. It is written in ANSI C.

The system consists of three main modules: a signal processing module that processes the waveform to produce feature vectors (Section 2), a neural network classifier that forms phonetic hypotheses (Section 3), and a dynamic programming module that scores the possible utterances (Section 4).

So far, *Hark* has been used to build two demonstration systems. One is an X11 window manager (Section 5). The other is an Army application that won't be described here.

For information about an integrated parser that is also being developed as part of the MITRE speech recognition work, see [1].

2 Signal Processing

The signal processing module produces a sequence of feature vectors to be classified by the neural network. The speech signal is sampled at 16 kHz and a feature vector is emitted every 10 ms.

The signal is bandpass filtered using a linear-phase finite impulse response filter with 217 taps. The filter has a minimum attenuation of 30 dB below 70 Hz to remove power line noise and DC bias in the A/D converter. The filter also attenuates frequencies above 6 kHz; the gain at 6.4 kHz is -6 dB.

Five scalar parameters and six spectral parameters are computed every 10 ms. The five scalar parameters are the fullband power, the power in the frequency band 0-800 Hz, the zero-crossing rate, the first reflection coefficient, and a measure of periodicity derived from the autocorrelation method of pitch period estimation. The six spectral parameters are the first six cepstral coefficients (not counting the zeroth term) derived from a sixth-order PLP analysis [2] which have been exponentially liftered [3]. These eleven numbers are stored in a frame buffer which contains values for the current frame and the

previous eight frames. The parameters from these nine frames are combined into a 73-dimensional feature vector which is handed to the neural network for a phonetic classification of the middle frame of the nine-frame block.

3 Neural Network Classifier

Hark uses the popular two-layer (one layer of *hidden* nodes and one layer of *output* nodes) feed-forward neural network trained on a large set of examples using the back-propagation training algorithm [4]. This type of network is also called a *multi-layer perceptron.*

The network has 73 inputs, 48 hidden nodes, and 41 output nodes. Each output node corresponds to a phonetic label. The labels used were drawn from the TIMIT label set [5, 6] and are listed in Table 1. To reduce the network size to 41 output nodes, we adopted the following conventions:

- /r/ and /axr/ were folded into /er/.
- All closures, silences, and pauses were folded into /cl/.
- /el/, /em/, /en/ and /eng/ were folded into /l/, /m/, /n/, and /ng/, respectively.
- /hv/ was folded into /hh/.
- /nx/ was folded into /n/.
- /y/ was folded into /iy/.
- /ux/ and /q/ were omitted.

The network was trained and implemented using MITRE's *Aspirin/MIGRAINES* neural network simulation package [7]. The training vectors were derived from 450 TIMIT sentences and 420 sentences that were recorded and transcribed jointly by MITRE and the Center for Spoken Language Understanding at the Oregon Graduate Institute of Science and Technology. The test set was derived from 120 TIMIT sentences and 180 MITRE/OGI sentences. A slight modification was made to the squared-error criterion so that nodes within 0.15 of their intended activation level were not penalized.

The network is used to do a *soft* classification, i.e., the network doesn't make a decision about the phonetic identity of the signal segment represented by the feature vector; rather, the entire vector of output activations is forwarded to the dynamic programming module. Even so, it is interesting to see how the network behaves as a classifier. The network makes the correct classification whenever the output node with the highest activation has the same label as the segment represented by the feature vector. For the TIMIT portion of the test set, the three highest scoring labels are /b/ (74.6%), /cl/ (73.8%), and /dx/ (72.3%), and the three lowest scoring labels are /ih/ (16.5%), /ah/ (16.6%), and /uw/ (20.5%). The average correct-classification rate is 42.2% with 11 labels scoring over 50%. At first, such figures would seem to predict poor performance in the dynamic programming module. But if we ask how often the node with highest *or second highest* activation has the correct label, then the average correct-classification rate jumps to 59.3% with 33 of the 41 labels scoring over 50%.

Table 1: Phonetic Labels for the 41 Output Nodes

Node	Label	Node	Label	Node	Label
1	/iy/	15	/oy/	29	/g/
2	/ih/	16	/ay/	30	/dx/
3	/ey/	17	/aw/	31	/jh/
4	/eh/	18	/l/	32	/p/
5	/ae/	19	/w/	33	/t/
6	/er/	20	/m/	34	/k/
7	/ix/	21	/n/	35	/ch/
8	/ax/	22	/ng/	36	/f/
9	/ah/	23	/v/	37	/th/
10	/uw/	24	/dh/	38	/s/
11	/uh/	25	/z/	39	/sh/
12	/ow/	26	/zh/	40	/hh/
13	/ao/	27	/b/	41	/cl/
14	/aa/	28	/d/		

4 Dynamic Programming

In order to decode utterances, *Hark* uses a dynamic programming algorithm. Specifically, we implemented an algorithm that is essentially the same as that described in Section III of [8]. *Hark* picks the best scoring utterance as its output; it does not make a list of multiple candidates.

Words are modeled as *pronunciation networks* [6]. These are directed graphs with nodes corresponding

to phonetic states. The simplest such model is a linear sequence of phonetic states. Examples for two words having such a model are shown in Figure 1. Some words will be pronounced differently depending on context, like "the." The model for "the" is shown in Figure 2; note that it has two different ending states. Finally, some words will be pronounced differently by different people, or pronounced differently by the same person in different situations. A model for the word "four" based on the pronunciations listed in *Webster's II New Riverside University Dictionary* is shown in Figure 3. Note that there are two possible paths through this network, representing different vowel qualities. Most words will require models with alternate pronunciation paths; however, they will usually be much more complicated than our model for "four" [6].

The set of recognizable utterances is also described by a directed graph called a *grammar network* [8]. A small portion of the grammar network for the X11 window manager (Section 5) is shown in Figure 4. An arc label in all lower-case letters corresponds to a specific word. An arc label in all upper-case letters corresponds to a set of words: TYPE refers to the set of words {Emacs, MATLAB, *Mathematica*, vi, xterm} (the first four are names of programs and the last is a generic window type) and DIGIT refers to the digits from one through nine. The label '#' is the *null* label, meaning that no word is used to make the transition. Examples of legal phrases for this portion of the grammar network include "the Emacs window," "window four", "four", etc.

Replacing the arcs of the grammar network by the individual word models yields a large directed graph. The goal of the dynamic programming algorithm is to find the optimal path through that graph given the sequence of neural network output vectors. The sequence of words corresponding to that best path is the decoded utterance.

As pointed out in [8], "Grammar nodes on a network allow paths reaching that node to merge so that only a limited number of the paths are allowed to grow to the succeeding grammar nodes." However, we have expanded the role of the grammar nodes; in order to handle pauses between words as well as continuous speech, each grammar node contains an internal directed graph that can be used to match silence.

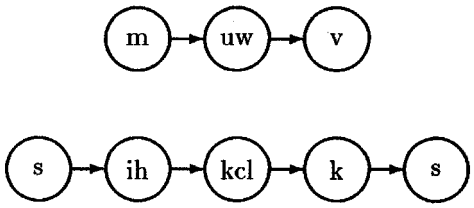

Figure 1: Digraphs for "move" and "six".

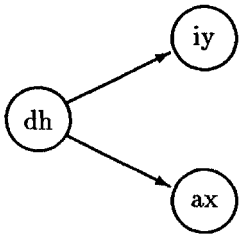

Figure 2: Digraph for "the."

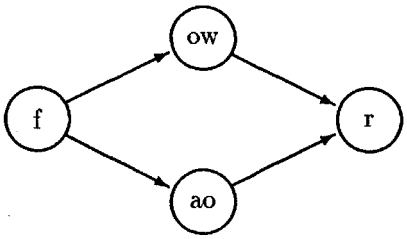

Figure 3: Digraph for "four."

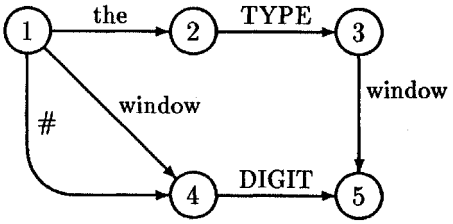

Figure 4: A portion of the grammar network for the X11 window manager.

5 An X11 Window Manager

In order to test and demonstrate the system, we developed a voice-controlled X11 window manager. The user can say things like "Open an emacs window," "Move the emacs window to the lower left," "Get a window," etc. There are 43 words in the vocabulary, and the application uses a simple command syntax that is modeled by a finite-state grammar network having 24 grammar nodes. This network generates 591 legal commands.

Performance statistics for the system are shown in Table 2 for five speakers who were *not* part of the neural network training or testing sets. The table shows the percentage of sentences that were recognized perfectly; the average correct-sentence percentage is 80.8%. Of the twenty-three sentences that were incorrect, sixteen were wrong because of a one word substitution error and two were wrong because of a one word deletion error. This version of *Hark* has performed quite well for three of the speakers, showing that it is somewhat speaker-independent. However, since it performed rather poorly for two of the five speakers, it is obvious that much work remains to be done.

Table 2: Window Manager Performance

Speaker	# correct / # sentences	Percentage Correct
fcct0	22/24	91.7
fdam0	21/24	87.5
fjmf0	14/24	58.3
mpes0	25/25	100.0
mrar0	16/24	66.7

6 Summary and Conclusions

We have described an experimental speech recognition system developed as part of MITRE's research program. The system has been used to build two application programs, one of which we have reported on here. We are now conducting research in order to improve the system. In addition, we are now working on integrating the system with the parser described in [1].

References

[1] T. Howells, D. Friedman, and M. Fanty, "Broca, an integrated parser for spoken language." International Conference on Spoken Language Processing, October 1992.

[2] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *J. Acoust. Soc. Am*, vol. 87, pp. 1738–1752, April 1990.

[3] J.-C. Junqua and H. Wakita, "A comparative study of cepstral lifters and distance measures for all pole models of speech in noise," in *Proceeding of the ICASSP*, pp. 476–479, 1989.

[4] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, vol. 4, pp. 4–22, April 1987.

[5] NIST, Speech Disc CD1-1.1 (Training and Test Data), *The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus.*

[6] K.-F. Lee, *Automatic Speech Recognition: the development of the SPHINX system.* Kluwer Academic, 1989.

[7] R. Leighton, "The Aspirin/MIGRAINES software tools, user's manual, release v5.0," Tech. Rep. MP-91W00050, The MITRE Corp., March 1992.

[8] C.-H. Lee and L. R. Rabiner, "A frame-synchronous network search algorthim for connected word recognition," *Trans. on ASSP*, vol. 37, pp. 1649–1658, November 1989.