

## DESIGN OF A RECONFIGURABLE STATE DECODING SYSTEM FOR HIDDEN MARKOV MODELS

José Gómez-Cipriano, Dante Barone, Sergio Bampi  
 Instituto de Informática – Universidade Federal do Rio Grande do Sul  
 Av. Bento Gonçalves, 9500 – Bairro Agronomia  
 Campus do Vale – Bloco IV  
 Caixa Postal 15064 – Porto Alegre – Brazil  
 Phone: 55-51-3167036 Fax: 55-51-3191576  
 Email: jgomez@inf.ufrgs.br

**Abstract** - In this work, the implementation of a Hidden Markov Model state decoding system is described. Hidden Markov Models are used in different kinds of sequence recognition problems. Specially, Hidden Markov Models are suited for speech/speaker recognition systems. The prototype was implemented for left-right Markov Models.

### 1. Introduction

Speech is a very natural and efficient way of communication for humans. The study of speech recognition concerns itself with the development of systems that emulate the human's ability to process such an exchange of information. Automatic speech recognition is the first step in a spoken language system that recognizes a person's word, interprets their meaning and provides an appropriate response. Research concerning spoken language systems is important since it will enable people to interact with machines using speech [1] [2].

In figure 1 [2], a block diagram of a canonic pattern recognition approach to speech/speaker recognition is shown. The pattern recognition paradigm has four steps, namely: (1) feature measurement, which is made on the input speech signal to define the "test pattern"; (2) pattern training, in which one or more test patterns corresponding to speech sounds of the same class are used to create a pattern representative of the features, producing a reference pattern or template; (3) pattern classification, in which the unknown test pattern is compared with each reference pattern and a measure of similarity between the test pattern and each reference is computed; (4) decision rule, in which the reference pattern similarity scores are used to decide which reference pattern best matches the unknown test pattern.

The factors that distinguish different pattern recognition approaches are the types of feature measurement, the choice of template or models for reference patterns, and the method used to create reference patterns and classify unknown test patterns.

A speech recognition system may work within certain, previously specified, constraints such as: size of vocabulary, speaker dependence or independence, manner of utterance (isolated vs. continuous) and response time. Depending upon these constraints, the computational load may be very high. If real-time characteristic is added as a new constraint, a possible solution could be to employ a specific purpose hardware.

In the approach shown in figure 1, speech is segmented into frames, which are time intervals of typically 10ms-30ms. The characteristics of every frame are described with a set of features where one feature could, for example, be a vector describing the energy of the speech signal in different frequency bands. The assumption is that these features were produced by a HMM speech process consisting of a finite set of N states and a set of transitions between these states.

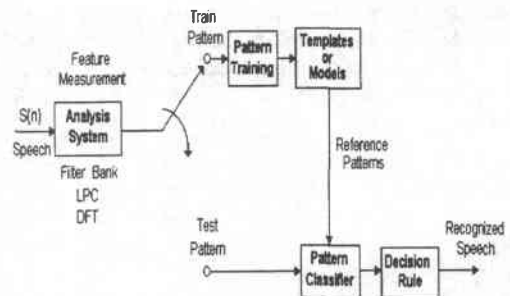


Fig. 1. Block Diagram of Pattern Recognition for Speech/Speaker Recognition [2].

Hidden Markov Models (HMMs) are used in the most successful modern speech/speaker recognition systems [1] [2].

A HMM speech recognition system consists of a probabilistic state machine and a method for tracing the state transitions of the machine for a given input speech waveform. The states in the HMM correspond to generic speech sounds (e.g. a group of phonemes, a phoneme or a part of a phoneme) where every state has a probability distribution that gives the output probability,  $P(o_i | q_i)$ , that the state  $q_i$  will output the

feature  $o_i$ . Speech can be considered as being generated by transitions between these states yielding a state sequence,  $q=(q_1 q_2 \dots q_T)$ . The likelihood of these transitions is described with a set of transition probabilities,  $a_{ij}$ , which reflect the probability that state  $S_j$  follows state  $S_i$ .

A HMM requires the observation probability distribution, that is the probability of observing the symbol  $k$  in state  $S_j$  and is denoted as  $b_j(k)$ . It becomes also necessary an initial state distribution probability  $\pi_i$  that gives the probability that the first state is the state  $S_i$ .

## 2. Using HMMs in Speech Recognition

Hidden Markov Models (HMMs) are the most widely used approach to speech recognition problems. The task of recognition is the problem of, given an observation sequence and HMM parameters, finding the most probable path or sequence of the hidden stochastic process. Our goal during speech/speaker recognition with HMMs is to trace out the most likely path through this state machine that could have produced the input speech waveform.

In a matrix form, the transition probability could be represented as:  $A=\{a_{ij}=P[q_{t+1}=S_j|q_t=S_i]\}$ . The observation probability could be represented in a matrix form as:  $B=\{b_j(k)=P[o_t=v_k|q_t=S_j]\}$ . The initial state distribution could be represented by the vector:  $\pi=(\pi_i=P[q_1=S_i])$ .

An HMM can be represented by using the compact notation  $\lambda=(A, B, \pi)$ . Specification of an HMM involves the choice of the number of states,  $N$ , the number of discrete symbols  $M$ , and three probability densities with matrix form  $A$ ,  $B$  and  $\pi$ .

For example, consider an isolated word recognizer based in HMMs. Assuming a vocabulary of  $v$  words to be recognized and that each word is to be modeled by a distinct HMM,  $\lambda_v$ . Further assuming that for each word in the vocabulary there is a training set of  $K$  occurrences of each spoken word (spoken by 1 or more talkers). Each occurrence of the word constitutes a observation sequence, where the observations are some appropriate representation of the (spectral and/or temporal) characteristics of the word. In order to do isolated word recognition, it must perform the following:

(1) For each word  $v$  in the vocabulary, the model parameters of the HMMs that optimize the likelihood of the training set observations for  $v$ th word must be estimated. This step is known as training.

(2) For each unknown word which is to be recognized, the processing of figure 2 must be carried out, namely measurement of the observation sequence  $O$ , via a feature analysis of the speech corresponding to the word; followed by calculation of model

likelihoods for all possible models,  $P(O|\lambda_v)$ . Finally, the selection of the word whose model likelihood is highest is realized [3] [4]. This step is known as recognition.

The recognition step could be divided in two fundamental blocks: The front-end parameterization and the recognition block. The front-end block operates over the input voice: A/D conversion, aliasing filtering, window-FFT, parameter extraction and vector quantization. The front-end output is a sequence of indexes. This sequence is the input for the recognition block. The recognition block makes the pattern training and the pattern matching. The heart of the recognition block is a state decoding system for the probability computation [5] [6]. The state decoding system must be able to decide, based on the input sequence coming from the front-end block, which is the most probable word uttered by the speaker.

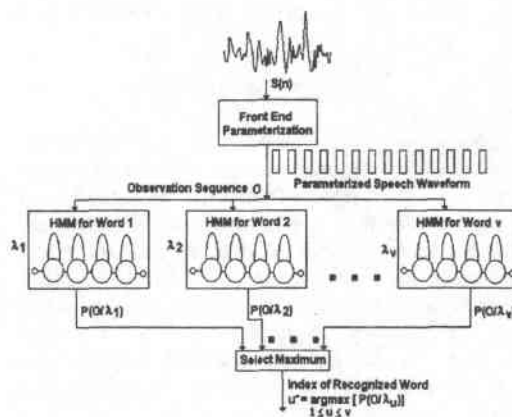


Fig. 2. Block Diagram of isolated word HMM Speech/Speaker Recognizer [2].

## 3. State Decoding

The state decoding problem consists in the determination of the state sequence  $q_i=(q_1 q_2 \dots q_T)$ , of the model  $\lambda$ , that most probably produce an observation sequence  $O=(o_1 o_2 \dots o_T)$ . By definition, in HMMs the state sequence is hidden. There is the possibility of reproducing the state sequence with the highest probably of generating the observation sequence [4] [7] [8].

For calculating the optimal state sequence it is necessary to calculate the highest probability inside a path, that ends in the  $i$ -state and produces the first  $t$  observations being in the time  $t$ . By defining the variable  $\delta_t(i)$  as the highest probability or best score of a single path which at time  $t$  ends in state  $i$  we have:

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} [P(q_1, q_2, \dots, q_{t-1}, q_t = S_i, o_1, o_2, \dots, o_t | \lambda)] \quad (1)$$

And by induction:

$$\delta_{t+1}(j) = \{ \max_i [\delta_t(i) a_{ij}] \} b_j(o_{t+1}) \quad (2)$$

Notice that equation (2) can be computed by only accessing the model parameters and the best scores for all the states at time  $t-1$ . This implies a frame synchronous progression of the computations, starting with  $t=1$  and continuing with one frame at a time, as time  $t$  evolves from  $1$  to  $T$ . The path information (i. e. the argument of the max operation in equation (2)) is stored in a  $T$  by  $N$  array  $\psi_t(j)$ , also called back-pointer array. The complete procedure to find the best state sequence will be:

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) \cdot a_{ij}] \quad (3)$$

The likelihood of producing the observation sequence given the model  $\lambda$ , will be:

$$P^*(O | \lambda) = \max_{1 \leq i \leq N} [\delta_T(i)] \quad (4)$$

The final state reached in the  $T$  interval of time is:

$$q_T = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \quad (5)$$

At the end of the sentence, after equation (5) has been calculated for the last frame, it is necessary a means of tracing the path from the state at the end of the most probable sequence back through the state trellis to the first frame, so that it could report the entire sequence. This is done by maintaining a backtrace list where, for every state and for every frame, a tag pointing to the most probable preceding state is stored. This tag is the address in the backtrace list of the preceding state. Using the tags, the most likely state sequence can be determined at the end of the sentence. The backtrace step will be:

$$q^*_t = \psi_{t+1}(q^*_{t+1}) \quad (6)$$

#### 4. Implementation

The topology of HMM is defined by the probable transitions. The general case where all the states are connected is known as ergodic model. A special topology widely used for speech recognition is the *left-right* model [5]. The state sequence of the *left-right* model has the property that the state index always increments or stays the same but never decrements in the time. In other words the state system goes from left to right.

In the figure 3, a *left-right* model is shown. Some simplifications could be made in the state-decoding block for left-right models. If  $j=1$  then  $\psi_t(j)=1$ , because if the next state is  $S_i=1$ , the current state must also be  $S_{i-1}=1$ . For the states  $j$  and  $i$ , the value of  $a_{ij}$  will not be zero only in the case that  $j=i$  or  $j=i+1$ .

The implementation of the state decoding system for the HMMs requires basic operations in the add-

compare-select block. In figure 4, this block adequated for speech recognition is shown. In order to minimize the amount of hardware needed to implement the state decoding block, we use the logarithm of probabilities. Thus, in the equations (2) and (3), all the multiplications are reduced to additions, and the maximum operation corresponds to a minimum operation since the absolute value of the logarithm of a high probability is a small number. The main computational load as described above, is in equation (2), and consists of  $N^2T$  add and maximize operations.

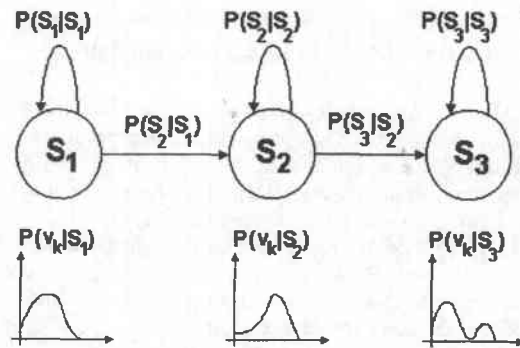


Fig. 3. Typical three-state HMM left-right model.

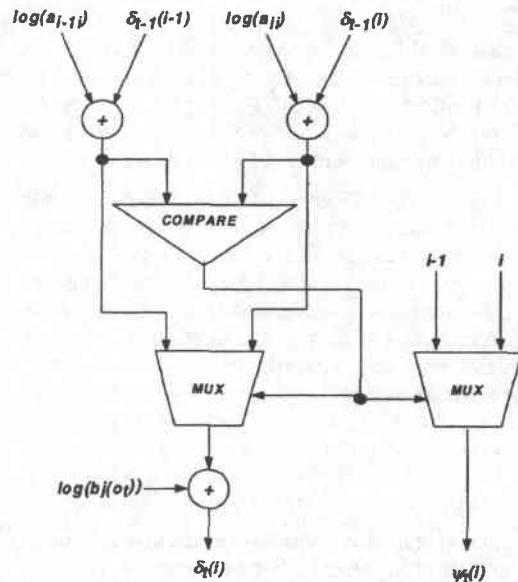


Fig. 4. Basic Block for add-compare-select operations.

Equations (2) and (3) above, are ideally suited for floating point representations, but such a representation would result in complex floating point datapaths and too large memories. To avoid this increased complexity, we implemented a fix-point

representation. For a fix-point implementation, we may employ a scaling procedure to maintain a limited range of the accumulated likelihoods  $\delta(i)$ .

In figure 5, the complete system is shown. A preliminary design of a state decoding for isolated speech recognition was implemented using FPGA. The Maxplus tool for Altera devices was used [9]. The circuit was mapped in the EPF10K20TC144-3 device. There were necessary 197 logic cells for an 8-state HMM state decoding. Simulation results showed a good performance.

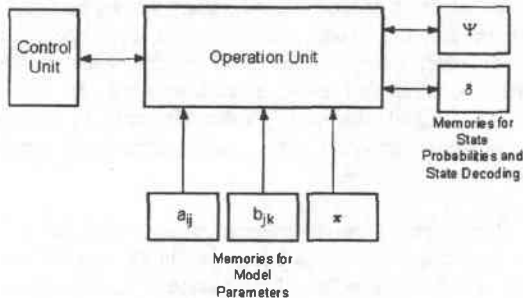


Fig. 5. State Decoding System.

### 3. Conclusions

This paper presents a preliminary design of a FPGA implementation of HMM state decoding.

The implementation is a starting point for a second-generation state decoder design that would serve as a practical building block for pattern recognition systems.

Future work involves the examination of the issues involved in integrating the state decoder into a computer system. The fix-point implementation makes necessary a stage normalization to retain accuracy and avoid underflow.

Perhaps the hardest task in the creation of an efficient high-level simulation model for the complete design that captures the non idealities of each block of a speech/speaker recognition system. The ultimate design goal of a speech recognition system is low recognition error. Achieving this goal requires circuit-level and microarchitecture optimizations.

### 4. References

- [1] LEE, K.- F. *Automatic Speech Recognition - The Development of the SPHINX System*. London: Kluwer Academic Publishers, 1989. 207p.
- [2] RABINER, L.; JUANG, B.- H. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.
- [3] CHONG, J.; TOGNERI, R. *Speaker Independent Recognition of Small Vocabulary*. *Sixth Australian International Conference on Speech Science and Technology*, Dec. 1996. p.133-137.
- [4] DELLER, J. R. et. al. *Discrete-Time Processing of Speech Signals*. Macmillan Publishing Co. 1993
- [5] MARKOWITZ, J. *Using Speech Recognition*. Prentice-Hall, 1996.
- [6] RABINER, L. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, vol. 77, No. 2, February, 1989. p. 257-286.
- [7] FOSTER, P.; SCHALK, T. B. *Speech Recognition. The Complete Reference Guide*. New York: Telecom Library, Inc., 1993. 377p.
- [8] TEBELSKIS, J. *Speech Recognition Using Neural Networks*. Pittsburgh: Carnegie Mellon University, 1995 Ph.D. Thesis.
- [9] SCARPINO, F. *VHDL and AHDL Digital System Implementation*. New Jersey: Prentice-Hall, 1998. 316p.