A Fully Parallel Mandarin Speech Recognition System with Very Large Vocabulary and Almost Unlimited Texts

Lin-shan Lee[1,2],Chiu-yu Tseng[3],Yueh Hong Lin[1],Yumin Lee[2],S.L.Tu[2],H.Y.Gu[1]
F.H.Liu[2],C.H.Chang[2],S.H.Hsieh[2],C.H.Chen[2],K.R.Huang[1],

[1]Department of Computer Science and Information Engineering, National Taiwan University,
[2]Department of Electrical Engineering, National Taiwan University,
[3]Institute of History and Philology, Academia Sinica,
Taipei, Taiwan, Republic of China

Abstract

This paper describes a fully parallel real-time Mandarin dictation machine which recognizes Mandarin speech with almost unlimited texts and very large vocabulary for the input of Chinese characters to computers. Isolated syllables including the tones are first recognized using specially trained hidden Markov models with special feature parameters, the exact characters are then identified from the syllables using a Markov Chinese language model, because every syllable can represent many different homonym characters. The real-time implementation is in occam language on a transputer system with 10 T800 processors operating in parallel. The overall correction rate for the final output characters is about 89%.

1 Introduction

Today, the input of Chinese characters into computers is still a very difficult and unsolved problem. All the currently existing input methods either are too slow or need special training, therefore can't be conveniently used by most people. This is the basic motivation for the development of a Mandarin dictation machine. We defined the scope of this research by the following limitations. The input speech is in the form of isolated syllables instead of continuous speech (the choice of syllables as the dictation unit will be discussed in detail later). The machine is speaker dependent. The first stage goal of this system is to have about 90% correction only for the sentence patterns in the Chinese textbooks of the primary schools in Taiwan, Rep. of China, because the errors can be found by the user on the screen and corrected from the keyboard very easily using convenient software system. Such a performance is still much more efficient than any of the currently existing input systems. However, on the other hand, the machine has to be able to recognize Mandarin speech with very large vocabulary (at least for sentence structures appearing in primary school text books) because we assume the input to computers can be arbitrary Chinese texts[1][2]. Also, the machine has to work in real-time for computer input applications. The above goals were almost achieved in a previous research[3]. This paper then presents a successfully implemented fully parallel version of the previously developed Mandarin dictation machine, but with significantly improved performance.

2 Considerations for Chinese and Overall System Structure

There are at least 60,000 commonly used words in Chinese. Therefore the words cannot be used as the dictation units. There are at least 15 thousands of commonly Chinese characters, each character is mono-syllabic. A nice feature is that the total number of different syllables in Mandarin speech is only about 1300. If we use the 1300 syllables as the dictation units, all the words or characters will be covered. However, the small number of syllables implies another difficult problem, that is, many different homonym characters will share the same syllable. This problem was then be solved by using a specially designed Markov Chinese language model. Based on the above observations on the special structure of Chinese language, the use of syllable as the dictation unit becomes a very natural choice.

Another very important feature of Mandarin is the lexical tones for the syllables. Mandarin Chinese is a tonal language. Every character is assigned a tone in general. There are basically five different tones. It has been shown that the primary difference for the tones is in the pitch contours, and the tones are essentially independent of the other acoustic properties of the syllables. If the differences among the syllables due to lexical tones are disregarded, only 408 syllables are required to represent all the pronunciations for Mandarin Chinese. This means the recognition of the syllables can be divided into two parallel procedures, the recognition of the tones, and the 408 syllables disregarding the tones.

Based on the considerations described above, the overall system structure for the fully parallel Mandarin dictation machine is shown in Fig.1. The system is basically divided into two subsystems. The

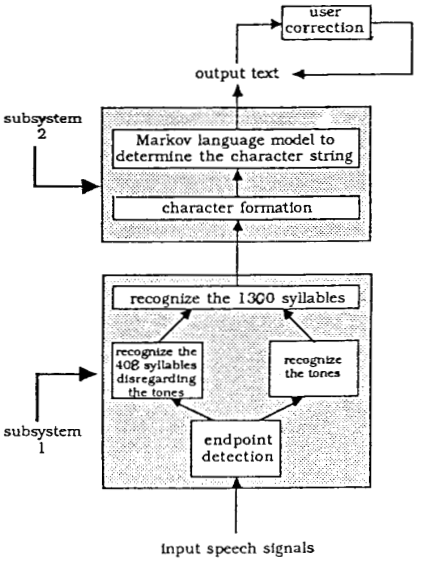


Figure 1: *The overall system structure for the Mandarin dictation machine*

first is to recognize the syllables, and the second is to transform the series of syllables into the characters. For the first subsystem of syllable recognition, the corresponding syllable (disregarding the tones) and the tone are then recognized independently in parallel. Because errors always happen, we therefore have to provide infor-

CH 3006-4/91/0000 – 0578 $1.00 © IEEE

mation for confusing syllables, and confusing tones. For the second subsystem we need to first to obtain all possible character hypothesis from a stored dictionary and then use the Markov Chinese language model to select the most probable (maximum likelihood) concatenation of them as the output sentence. All these processes will be described in detail in the following.

3 Syllable Recognition Disregarding the Tones

The recognition of the 408 Mandarin syllables disregarding the tones is very difficult because there exist 38 confusing sets in the vocabulary, each of which contains at most 19 very confusing syllables. Conventionally each Mandarin syllable is decomposed into an "initial/final" format, in which "initial" means the initial consonant of the syllable, while "final" means the vowel or diphthong part but including possible medial or nasal ending. Each confusing set mentioned above then consist of syllables sharing the same final but with different initials. Direct application of standard approaches of hidden Markov models (HMM's)[4] gives recognition rates on the order of only 70% ~ 80% due to the difficulties caused by these confusing sets. A revised three-pass training approach for HMM's has been developed by specially considering the characteristics of this vocabulary. Because all the syllables in a confusing set share the same final, in this approach 38 final HMM's are first trained in the first pass, each for the final of a confusing set using the segmented final parts of one set of the training utterances, and 99 initial HMM's are then trained using the segmented initial parts in the second pass. These initial and final HMM's are finally smoothly cascaded to form 408 syllable HMM's by requiring that in each syllable HMM the last state of the initial HMM is exactly the first state of the final HMM which was trained primarily from the transition region of the speech signal. In this way not only the initial HMM's and final HMM's can be separately trained and the short initial parts can be assigned more number of states, but the HMM's for the syllables in a given confusing set will have exactly identical parameters in the last few states, thus the effect of the final in the recognition phase can be minimized while differences in initials can be emphasized to better distinguish these syllables. The 99 initial models are obtained as follows. Considering the fact that very often quite a few finals approximately start with some common phoneme (for example, [a], [ai], [au], [an], [ang] all start with the phoneme /a/), syllables with these finals but the same initial(such as [sa], [sai], [sau], [san], [sang]) can in fact share the same initial HMM. In this way, the total number of initial HMM's was found to be 99. In fact, only one set of training utterances needs to be segmented into initials and finals, and they are used to train 38 final HMM's and 99 initial HMM's to be cascaded to form the 408 syllable HMM's. These 408 HMM's are then taken as the initial values to go through a third pass training, in which the unsegmented training utterances are used in the forward-backward algorithm, while the parameters of the initial and final parts of the syllable HMM's are reestimated separately. Simulation results indicate that in this way the top 1 recognition rate can be as high as 91.42% as compared to the results of 70% ~ 80% for the standard HMM's.

4 Fast Approaches for Syllable Recognition

The difficulties in real-time implementation of the syllable recognition is due primarily to the very large amount of computation:

$$\begin{aligned} &75 \text{ frames} \times 408 \text{ syllables} \times 7 \text{ states} \times [5 \text{ mixs} \times \\ &11 \text{ dims} \times (1 \text{ mult} + 1 \text{ add}) + (1 \text{ exp} + \\ &4 \text{ subtracts}) + (3 \text{ mults} + 1 \text{ add})] \\ &= 25,275,600 \text{ float mults} \end{aligned}$$

Here one addition is regarded as 1/2 multiplication and one EXP is regarded as 30 multiplications according to the floating point performance of INMOS transputer T800-20 and the occam arithmetic library supported in INMOS TDS (Transputer Development System). It takes at least 16 sec to recognize one syllable using one 1.5 MFlops/sec processor. Therefore 30 such processors are needed to compute it in real time. Several approaches were therefore developed to reduce the computation requirements.

First, since the models representing the syllables with the same initial share the same parameters of the first few states and the models representing the syllables with the same final share the same parameters in the last few states, we can thus combine the computation for the shared parameters together. This reduces the necessary computation by a factor of 6. However, more program overheads are needed to handle the combination of shared computation. Another approach to reduce computation is to evaluate all models at first, but delete the less possible models soon. A useful method called two-stage search was developed. In the first stage, 38 syllables with 38 different finals are recognized. In the second stage, only those syllable models consisting of the k most possible finals found in the first stage are recognized, because the final part of Mandarin syllables are very stable and easy to distinguish. This approach can speed up the computation by 3 times with a 0.49% loss only in recognition rate when $k = 3$. In fact, if we make a rough cut between the initial and final and simply take 3/4 or 4/5 of an utterance as the final part to be used in the first stage, the recognition rate will be even higher. However, because in such an approach the second stage cannot be started until the complete syllable is uttered, a significant delay always exists no matter how efficiently it can reduce the computation. On the other hand, a more straight forward approach to reduce computation is the well-known beam search. This is to recognize an utterance with all 408 syllable models at first. After evaluation of the first few frames, we select only the most possible syllable models as candidates of the utterance and continue to evaluate their probabilities for the successive frames. Such selection can be done repeatedly. The probability of the model for the correct syllable usually becomes significantly higher than most of the other 407 syllable models very soon. This makes the beam search approach very attractive. It is easy to decide a sequence of phases and beam width for beam search which can speed up the computation several times but with almost no loss in recognition rate. In fact, some utterances can be correctly recognized in beam search while missed in full search. This is because most confusing syllable models have the same final but different initial, thus are easily deleted in the beginning frames while become very confusing with similar probabilities after all the frames are evaluated. In fact, the two-stage search and the beam search approaches can be used together, in which beam search can be applied in the first stage and/or the second stage of the two-stage search.

5 Implementation of the Syllable Recognition

Now, we consider how to parallelize the syllable recognition in many processors. The available processors are the INMOS transputers T800-20, a 32 bit, 10 MIPS processor with integral 64 bit floating point unit with sustained performance of 1.5 MFlops/sec. The T800 uses a DMA block transfer mechanism to transfer messages with other transputers via four INMOS links. The link interface and the processor all operate concurrently, allowing processing to continue while data are being transferred on all of the links. The four serial links on the T800 give a unidirectional transmitted rate of 1.7 Mbytes/sec and a combined (bidirectional) data rate per link of 2.3 Mbytes.sec, at a link speed of 20 Mbits/sec.

To parallelize the syllable recognition, we must distribute its computation to many processors first. There are many factors involved for the total computation amount. These include the number of frames, number of syllable models, number of states, number of mixtures and order of autocorrelation. It is difficult (but not impossible) to distribute the computation according to number of

frames because for real-time processing, the input frames are produced one after one and all computation must follow this sequence. For example, we cannot finish the computation about the second frame before the results about the first frame are available. Distributing the computation according to syllable models is ideal if we don't combine the duplicated computation for shared parameters together as mentioned above, in which, the evaluation for every syllable model uses the same input but is completely independent except that their results must be collected and compared. Once the duplicated computation are combined, an approach is to partition the syllable models according the cross relationships in the parameter-sharing. The 408 syllables can be partitioned into seven groups (as in Table 1).

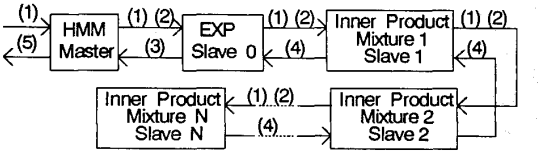
group	final
*	null final
a	a, ai, au, an, ang
o	o, ou
e	e, eh, ei, en, eng, er
i	i, ia, ie, iai, iau, iou, ian, in, iang, ing
u	u, ua, uo, uai, uei, uan, uen, uang, ueng
iu	iu, iue, iuan, iun, iung

Table 1: *Syllable partition table according to parameter-sharing relations*

In this way, a syllable shares either the same initial HMM or the same final HMM with some other syllables belonging to the same group but shares no parameter with any syllable in other groups. To distribute the computation according to syllable models can then refer to this partition to minimize data dependence among processors. Because the number of initials or number of finals contained in every group here are not equal, more processors should be assigned for larger group to balance the load. Such arrangement seems reasonable, but becomes infeasible if the two-stage search mentioned above were applied or turns out to be unbalanced if beam search were used. Better dynamic allocation mechanism is thus needed.

An alternative is to distribute the computation according to the states of HMM's. In this case every processor shares the same input and should pass the transition probabilities to the processor which evaluates the probabilities for the next state. The data dependence flow is simple and unidirectional. However, the computation necessary for each state is not equal because there are 99 initials modeled using the first few states but 38 finals modeled using the last few states. Even though we can assign more processors for the state with more computation involved, once the two-stage search or beam search is applied, we still need a dynamic allocation mechanism to balance the load among processors.

We can also consider allocating the computation according to the mixture index. The syllable recognition evaluation can be regarded as two parts, calculating the observation probabilities and the state transition probabilities respectively. The former takes more than 90% of the total computation. We can thus assign one processor as the master to calculate the state transition probabilities and other processors as slaves to calculate the observation probabilities. Distribution of the computation for these slave processors is then based on the mixture index. Every slave processor can be responsible for the inner product of the input autocorrelation vectors with the characteristic vectors of a certain mixture. They share the same input and their results should be merged and compared to choose the largest one. The exponential value of the largest one is then computed as the observation probability and sent to the master processor. The structure is shown in Fig.2. Although the inner product and PDF values must be passed among processors, this won't increase too much overall computation time because the links of T800's have their own DMA interfaces to transfer these data concurrently with the processors' operation. The data dependence flow is simple and unidirectional. There are at least three important advantages in this configuration. First, the computation load is always balanced no matter the two-stage search or beam search is applied. Secondly, it will be easy to modify the connection relations among the HMM states or the observation probability function, because the two parts are separated in this configuration. Lastly, different specific functions (e.g. HMM state transition probabilities, inner product or exponential value evaluation) are separately implemented, thus it will be easy to implement this configuration on specially designed circuits with low cost. This is why we finally configure the syllable recognition subsystem in this structure. There still exists one more different approach, i.e., to distribute the computations according to the order of autocorrelation. However, this will cause too much data dependence and communication because every product obtained in one processor must be added to another product obtained in another processor. Note that a 32 bit floating point multiplication takes the processor only 650 ns, but 2350 ns will needed for the link to pass the 32 bit result to another processor.



(1) Input autocorrelation vectors
(2) Selected candidates by fast method or beam search approach
(3) Pdf value of input autocorrelation vector
(4) Maximum inner product value of mixture 1 to mixture N
(5) Most possible syllables for input utterance

Figure 2: *Block diagram of syllable recognition sub-system*

6 Tone Recognition

There are totally five different lexical tones in Mandarin usually referred as the the first, second, third, fourth and the fifth tones, among them the fifth(or neutral) tones is the most difficult to distinguish. This is because there exist typical pitch contours for the pitch frequencies of the first four tones in general. However, the pitch frequencies of the fifth tone do not necessarily follow any specific pattern. Some initial efforts on Mandarin lexical tone recognition had been reported with very encouraging results[5], but they all concentrated on the recognition of the first four tones while the fifth tone was always ignored. In this research,special algorithm is used to distinguish the fifth tone, and we give up the conventional SIFT algorithm but choose the improved parallel processing approach to detect the pitch interval in order to reduce the computation requirements, in which only one pitch-period estimator is used. The block diagram is as shown in Fig.3(a). First, the speech signal is lowpass filtered with a cutoff of about 600 Hz to produce a relatively smooth waveform. Following the filtering, the "peaks and valleys" (local maxima and minima) are located. An impulse train is then derived from the filtered signal. Each impulse has a value equal to the difference between the peak amplitude and preceding valley amplitude, and is located at each peak. The impulse train is next processed by a time varying nonlinear operation. When an impulse is detected in the input, the output is set to the value of that impulse and then held for a blanking interval, T_a, during which no pulse can be detected. At the end of the blanking interval, the output begins to decay and then rise exponentially. When an impulse exceeds the level of the output, the process is repeated. The duration needed to detect the next impulse is an estimate of the pitch period. The nonlinear time function is shown in Fig.3(b).

The extracted pitch periods are first smoothed using a median filter. The observation vector sequence is then derived from this smoothed pitch period. We use the conventional feature vector

definition for tone recognition,

$$\mathbf{X}_t = [\log(f_t) + \log(f_{t-1}), \log(f_t) - \log(f_{t-1})]$$

where f_t is the pitch frequency at time t. In order to improve the recognition of the fifth (neutral) tone, an additional parameter, the probability of the syllable duration, is used to evaluate the final probability.

$$P_{final} = P_{dura} \times P_{hmm}$$

where P_{dura} and P_{hmm} are respectively the probability of the syllable duration and the discrete HMM with codebook size 16. P_{dura} can be estimated from the distribution of the training data. The test results indicate that the recognition rate of all the five tones is 92.3%.

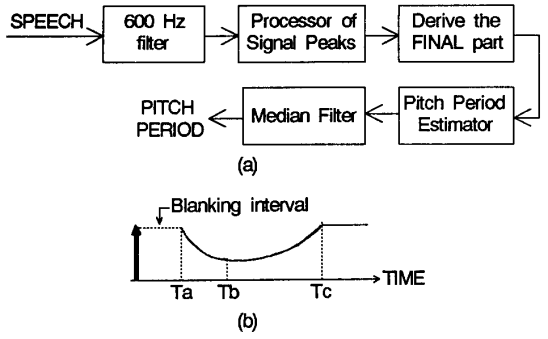


Figure 3: *(a) Block diagram of PPE (b) Nonlinear time function*

7 The Markov Chinese Language Model

Even if the syllable and tones can be correctly recognized using speech processing approaches, the high degree of ambiguities due to homonym characters still causes serious problems for the linguistic decoding process in the Mandarin dictation machine. This is because each Chinese character is pronounced as a monosyllable and in average each phonologically allowed syllable can represent at least 15 homonym characters. Thus the choice of the correct character for each syllable is a very difficult task.

In this research, a powerful Markov model for Chinese language was developed to solve this difficult linguistic decoding problem[6]. The likelihood value for a given Chinese sentence is the product of the successive state transition probabilities, where the states can be either characters or words (a word is composed of one to several characters). This is because unlike English language there is no boundary marker between two adjacent words. The state transition probabilities are trained using a large quantity of training texts. For each sequence of input candidate syllables and probabilities, a word lattice can be constructed and the most probable output Chinese sentence can be obtained from the maximum likelihood path found in this lattice. Although the number of possible paths in the lattice is of exponential size in general, an efficient search algorithm based on dynamic programming was also developed to find the best solution in polynomial time.

Our language model is to compute the Markov chain probabilities of the possible output sentences for the input sequences of syllables. Apparently, the computation amount will increase exponentially with the length of the sequence. Beam search thus must be imposed to avoid such increase. Experimental results show that the model performs well under beam search imposed, and the evaluation time can be acceptable. The parallelization strategy is straight forward. Suppose there are N characters corresponding to a syllable, numbered 0 to $N-1$, and we have M processors available, numbered 0 to $M-1$. We allocate the computation about those characters whose number is i mod M to processor numbered i. This allocation makes the load well balanced. Another important advantage in this configuration is that not only jobs are equally distributed among the processors, but the large database storing the statistical relationships among the characters is also evenly divided for every processor.

8 Physical Structure and Preliminary Test Results

The block diagram of the fully parallel machine is shown in Fig.4. An IBM PC/AT is used as I/O controller for keyboard, monitor and disk driver. An A/D converter is directly connected to the host transputer to sample the filtered input utterances. The host transputer runs the user interface routine and the end point detection procedure. The input utterance is then sent to the next transputer responsible for tone recognition, concurrently a bypass process passes the input samples to the next transputer responsible for calculating the autocorrelation vectors. The autocorrelation vectors are then sent to the syllable recognition subsystem which consists of seven transputers, with configuration discussed previously. Since the language model subsystem cannot be started until the recognition units finished their jobs, it can share the same processors with the syllable and tone recognition subsystems. In other words, after syllable and tone recognitions are finished, the language model is run on the first eight transputers. There are therefore a total of ten transputers used in the machine. Preliminary tests indicate that the time needed for the machine to recognize the syllable and tone is shorter than the time needed to produce the utterance and there is less than 0.01 second delay between the speaker finishes an utterance and the estimated syllable and tone are decided. The time needed for the language model to process fifteen syllable hypotheses to obtain fifteen most probable characters is about 0.2 second. The overall correction rate for the final output characters is found to be about 89% in the preliminary test.

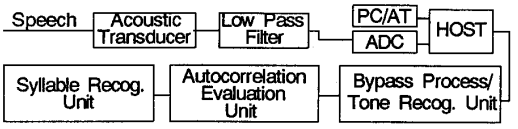


Figure 4: *Block diagram of overall system*

9 Conclusion

A fully parallel real-time Mandarin dictation machine which recognized Mandarin speech with almost unlimited texts and very large vocabulary has been completed on a transputer system. Isolated syllables are recognized using hidden Markov model techniques and transformed into Chinese characters through a Markov Chinese language model under speaker dependent mode. The overall system performance is still under test. The technology used in this machine is quite different from those machine for other language due to the very special characteristics of Chinese language.

References

[1] L. R. Bahl et. al., "Large Vocabulary Natural Language Continuous Speech Recognition", ICASSP-89, pp.465-467

[2] B. Merialdo, "Multilevel Decoding for Very-large-size Dictionary Speech Recognition", IBM J. Res. Develop., Vol. 32(2), pp.227-237, Mar., 1988.

[3] L.-S. Lee et. al., "A Real-Time Mandarin Dictation Machine for Chinese Language with Unlimited Texts and Very Large Vocabulary", ICASSP-90, pp.65-68.

[4] B.-H. Juang and L. R. Rabiner, "Mixture Autoregressive Hidden Markov Models for Speech Signals", *IEEE Trans. ASSP*, pp.1404-1413, 1985.

[5] W. J. Yang et. al., "Hidden Markov Model for Mandarin Lexical Tone Recognition", *IEEE Trans. ASSP*, Vol. 36, pp.988-992, Jul., 1988.

[6] S. M. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer", *IEEE Trans. ASSP*, Vol. 35(3), pp.400-401, Mar., 1987.

Acknowledgements

The work described in this paper is supported by the Nation Science Council of Taiwan, Republic of China from 1984 to 1990.