Large Vocabulary Word Recognition based on a Graph-Structured Dictionary

Katsuki Minamino

D21 Laboratory, SONY Corporation
6-7-35, Kitashinagawa, Shinagawa-ku, Tokyo, 141 Japan

ABSTRACT

In this paper, a structural search using a word-node graph is proposed to speed up the isolated word recognition based on hidden markov models (HMMs). We define a distance measure for comparing pairs of words, and construct a graph keeping the structure of word distribution. Based on this graph, the number of words to be examined are restricted. Experiments show that the search complexity is considerably reduced with little degradation of the recognition accuracy.

1. Introduction

In isolated word recognition [1], the goal is to find the word w_j maximizing the probability $P(w_j|y)$ for an observed sequence y. This is, in general, performed by using the phoneme based HMMs, a lexical tree, and the Viterbi algorithm [2]. The computational load has two major parts : local likelihood calculation, and trellis search. If a continuous distribution is used for the output probability, it is important to reduce the first part, that is, the likelihood calculation for all the probability density functions. For this problem, an approach using a tree-structured probability density function [3], or vector quantization [4] was proposed.

As for the second part, if full search is used, the complexity increases linearly with the vocabulary size. It is, therefore, necessary to reduce the search complexity for large vocabulary speech recognition in a real-time system. Beam search is a well-known technique. However it has the problem that search errors are caused by pruning in excess. Another approach is A^* search (e.g., a tree-trellis search [5]). The efficiency of this technique is dependent on the heuristics. The other approach is that we preliminarily select words to search, and replace the problem by small vocabulary word recognition. In this approach, it is necessary not to leave out the words w_j with high probability $P(w_j|y)$.

In this paper, we propose new search algorithm using a word-node graph. This is a word pre-selection technique based on the word distance measure. To put it shortly, given a sequence y, we first calculate the probability $P(w_j|y)$ for some words w_j; secondly, we calculate the probability $P(w_i|y)$ of the words w_i that are similar to the words w_j with high probability. For this purpose, we define a distance measure between comparing pairs of words, and introduce a space for words based on the distance. Next, we construct a directed graph representing word distribution, where the nodes correspond to words. In the recognition process, the graph is used for the word selection, so that the search space is restricted.

In section 2, we define a new distance measure between word pairs. In section 3, we describe the method of constructing the word-node graph. In section 4, we present the recognition procedure using this graph. In section 5, experimental results are shown. Finally, we summarize our conclusion.

2. Word Distance Measure

2.1. Definition

In this section, the distance measure between word pairs is defined for the sake of representing word distribution. We consider a word set $\{w_1, w_2, \ldots, w_N\}$ of the vocabulary size N. Let Y_j be the observation symbol sequence set corresponding to the word w_j. The parameters of HMMs are generally trained to maximize the log likelihood $\sum_{k=1}^{n} \log P(y_j^k|w_j)$ for the data set $\{y_j^k \in Y_j \mid k = 1, \ldots, n\}$. As a result, we obtain the word model $\hat{w}_j$ corresponding to each word w_j. Then, the recognition process is carried out by calculation of model likelihoods $\log P(y|\hat{w}_j)$, $1 \leq j \leq N$.

Now we consider the expectation of $\log P(y_i|\hat{w}_j)$ under the fixed model $\hat{w}_j$, i.e.,

$$E[\log P(y_i|\hat{w}_j)] = \int P(y|w_i) \log P(y|\hat{w}_j) dy, \qquad (1)$$

where $E[\ ]$ is the expectation operator, and y_i is the observation symbol sequence corresponding to the word w_i. This is approximately computed by using the observed data set $\{y_i^q \in Y_i \mid q = 1, \ldots, Q\}$ as follows:

$$E[\log P(y_i|\hat{w}_j)] \approx \frac{1}{Q} \sum_{q=1}^{Q} \log P(y_i^q|\hat{w}_j). \qquad (2)$$

If the model parameters are well estimated, then

$$E[\log P(y_i|\hat{w}_i)] \geq E[\log P(y_i|\hat{w}_j)] \tag{3}$$

with equality if and only if $\hat{w}_i = \hat{w}_j$. Furthermore, it seems reasonable to suppose that if $E[\log P(y_i|\hat{w}_j)]$ is large, then $\log P(y|\hat{w}_j)$ may be also large for $y \in Y_i$.

Therefore, we define the distance measure D from w_i to w_j by

$$D(w_i, w_j) := E[\log P(y_i|\hat{w}_i)] - E[\log P(y_i|\hat{w}_j)]. \tag{4}$$

The distance measure D has two properties:

$$\begin{aligned} &\text{(i)} \quad D(w_i, w_j) \geq 0. \\ &\text{(ii)} \quad D(w_i, w_j) = 0 \quad \Leftrightarrow \quad w_i = w_j. \end{aligned} \tag{5}$$

We should notice that this measure is directed, relative, and non-symmetric. If the distance $D(w_i, w_j)$ is small, then $E[\log P(y_i|\hat{w}_j)]$ is as large as $E[\log P(y_i|\hat{w}_i)]$. This implies that the word w_j may have high probability $P(w_j|y)$ for $y \in Y_i$.

Next, we shall focus on the calculation of the distance D. The calculation of $E[\log P(y_i|\hat{w}_j)]$ using the equation (2) needs the observed sequence set $\{y_i^q \in Y_i \mid q = 1, \ldots, Q\}$. However, in large vocabulary word recognition using phoneme-based HMMs, such a sequence set $\{y_i^q\}$ is not always available in practice. We wish, therefore, to calculate the expected value $E[\log P(y_i|\hat{w}_j)]$ by using only the model parameters.

2.2. Approximation

We assume that there exists a typical symbol sequence $E[y_i]$, which corresponds to the word w_i, satisfying the approximation

$$E[\log P(y_i|\hat{w}_j)] \approx \log P(E[y_i]|\hat{w}_j). \tag{6}$$

Our concern is to generate such a sequence $E[y_i]$ based on HMM parameters.

Assume that each HMM is a left-to-right model with no skipped state. Let a_{ij} be a state transition probability, and $b_{ij}(k)$ an observation symbol probability. Based on a_{ij}, the probability of duration n in state i is given by $(a_{ii})^{n-1}(1 - a_{ii})$. Thus the expectation of n is

$$E[n] = \sum_{n=1}^{\infty} n(a_{ii})^{n-1}(1 - a_{ii}) = \frac{1}{1 - a_{ii}}. \tag{7}$$

Based on the expected value $E[n]$, the state transition sequence $X_i = x_0 x_1 x_2 \cdots x_T$ is uniquely decided for the word w_i, where T is the length of the sequence X_i. If the symbol sequence $Z_i = z_1 z_2 \cdots z_T$ is generated according to the state transition sequence X_i, then the probability $P(Z_i|\hat{w}_i, X_i)$ is given by

$$\begin{aligned} P(Z_i|\hat{w}_i, X_i) = a_{x_0 x_1} b_{x_0 x_1}(z_1) \cdot a_{x_1 x_2} b_{x_1 x_2}(z_2) \\ \cdots a_{x_{T-1} x_T} b_{x_{T-1} x_T}(z_T). \end{aligned} \tag{8}$$

Now we generate the symbol sequence Z_i such that the probability $P(Z_i|\hat{w}_i, X_i)$ is maximized. It is easily decided as the sequence of symbol z_t maximizing each probability $b_{x_{t-1} x_t}(z_t)$. If the observation symbol probability distribution is a Gaussian distribution, then the symbol z_t is uniquely decided as an expected symbol. Moreover, the sequence X_i is an expected sequence based on the state transition probability. Therefore, we use the symbol sequence Z_i for the new definition of word distance measure $\bar{D}$:

$$\bar{D}(w_i, w_j) := \log P(Z_i|\hat{w}_i) - \log P(Z_i|\hat{w}_j), \tag{9}$$

where Z_i is defined by

$$Z_i := \underset{Z_i}{\operatorname{argmax}} \; P(Z_i|\hat{w}_i, X_i). \tag{10}$$

In the definition of (9), the probability $P(Z_i|\hat{w}_j)$ is computed with the scoring procedure (e.g., the forward algorithm, or the Viterbi algorithm). If the approximation $E[\log P(y_i|\hat{w}_j)] \approx \log P(Z_i|\hat{w}_j)$ is warrantable, the distance measure $\bar{D}$ has similar properties to the distance measure D of (4).

2.3. Normalization

The distance measure $\bar{D}(w_i, w_j)$ is highly dependent on the length of the symbol sequence Z_i generated for the word w_i. It is important, therefore, to normalize the distance measure $\bar{D}(w_i, w_j)$. One approach is to normalize the term $\log P(Z_i|\hat{w}_j)$ in the definition (9) by

$$\log P(Z_i|\hat{w}_j) := \frac{\log P(Z_i|\hat{w}_j)}{T}, \tag{11}$$

where T is the length of the sequence Z_i. Another approach is by

$$\log P(Z_i|\hat{w}_j) := \frac{\log P(Z_i|\hat{w}_j)}{\sum_{j=1}^{N} \log P(Z_i|\hat{w}_j)}, \tag{12}$$

such that $\sum_{j=1}^{N} \log P(Z_i|\hat{w}_j) = 1$. As the other approach, we propose the normalized distance measure as follows:

$$\tilde{D}(w_i, w_j) := \mathrm{Order}[w_j|\bar{D}(w_i, w_j)], \tag{13}$$

where Order[ ] is the mapping to the order of word w_j according to the distance $\bar{D}(w_i, w_j)$. To sum up, we intend to normalize the distance $\bar{D}$ of (9) by the mapping into the integer $0, 1, \ldots, N-1$. By these normalization, we can evaluate all distance $D(w_i, w_j)$, $(1 \leq i \leq N, 1 \leq j \leq N)$, based on the same criterion.

As the definition of word distance measure, Juang and Rabiner [6] have proposed a probablistic distance measure for HMMs,

$$D(\lambda_i, \lambda_j) = \lim_{T \to \infty} \frac{1}{T} \{\log P(O_T|\lambda_i) - \log P(O_T|\lambda_j)\}. \tag{14}$$

Furthermore, the modified distance measure

$$D(\lambda_i, \lambda_j) = \frac{1}{N_i} \sum_{k=1}^{N_i} \{\log P(A_{ik}|\lambda_i) - \log P(A_{ik}|\lambda_j)\} \quad (15)$$

is used by D'Orta, Ferretti, and Scarci [7]. Here the symbol sequence O_T, or $A_{ik}, (k = 1, \ldots, N_i)$ is generated according to the state transition probability and the observation symbol probability of the model λ_i. But they are randomly generated. This is a different point from our definition (9).

3. Word-node Graph

Based on the distance measure $D(w_i, w_j)$, all words $w_j, (j = 1, \ldots, N)$ are distributed over a space with word w_i as the origin. It must be noted that no metric space is defined for words, because the axiom for metric is not satisfied by the distance measure $D(w_i, w_j)$.

Now, we construct a graph keeping the structure of word distribution. The procedure is as follows:

1. **Initialization :** Define the node set
$$G_0 := \{w_1, w_2, \ldots, w_N\}, \quad (16)$$
where the nodes correspond to words. Set n = 0.
2. **Word Connection :** For the node set G_n, connect the word pairs w_i and w_j satisfying
$$D(w_i, w_j) \leq r_a \quad \text{and} \quad D(w_j, w_i) \leq r_a, \quad (17)$$
where r_a is a certain threshold. Then we get some connected graphs including simple node graphs.
3. **Representative Selection :** For each connected graph in step 2, select the node having maximal connection, as the representation. Next, remove these representative nodes and other nodes connected with the representatives, so that some subgraphs remain. Again, select the representative nodes from these subgraphs. This process is repeated until all nodes are removed. As a result, G_n is divided into representative nodes and non-representative nodes.
4. **Bipartite Graph Expansion:** Define G_{n+1} as the representative node set of G_n. Set the directed path from the node of G_{n+1} to the node of G_n according to the connection in step 2. Also, set the directed path from the node of G_{n+1} to the self-same node of G_n. Thus we get the bipartite graph, where all nodes of G_n are connected from at least one node of G_{n+1}.
5. **Path Addition :** For all nodes w_i of G_n, set the directed path from all nodes w_j of G_{n+1} satisfying
$$D(w_i, w_j) \leq r_b, \quad (18)$$
where r_b is a certain threshold.
6. **Completion :** Replace n by $n + 1$. If the element number of G_n is less than a certain threshold, then halt. Otherwise, return to step 2.

We get $G_0, G_1, \ldots, G_n$ in turn. The directed paths are connected from the nodes of G_{n+1} to that of G_n. Step 2-5 are repeated using only the representative nodes from the current layer. After all, we obtain a layered, hierarchical, directed graph. All nodes correspond to words, and the leaves, the nodes of G_0, correspond to the vocabulary. We call this graph a **word-node graph**.

From (17) and (18), it follows that

$$D(w_i, w_j) \leq r_a \quad \text{or} \quad D(w_i, w_j) \leq r_b \quad (19)$$

for all pairs of child node w_i and parent node w_j on the word-node graph. It means that the expectation $E[\log P(y_i|\hat{w}_j)]$ is as large as $E[\log P(y_i|\hat{w}_i)]$. Therefore, if the word w_j in the n-th layer has high probability $P(w_j|y)$ for a given sequence y, then we can hypothesize as follows:

- y is the symbol sequence corresponding to the word w_j.
- y is the symbol sequence corresponding to the child word w_i, connected from w_j, in the $(n-1)$-th layer.

Based on these hypotheses, we realize a recognition method using the word-node graph.

A vocabulary of words to be recognized is called a dictionary. And the combination of dictionary and the corresponding word-node graph is called a **graph-structured dictionary**.

4. New Search Algorithm

In this section, we present new search technique using a graph-structured dictionary. Define the dictionary

$$W := \{w_1, w_2, \cdots, w_N\}, \quad (20)$$

where w_i is word, and N is the vocabulary size. After building an HMM for each word w_i, a word-node graph for the dictionary W is designed beforehand using only the HMM parameters. The bottom layer corresponds to the dictionary. If we have the word-node graph of L-layers, the search procedure is realized as follows:

1. **Initial Search :** Given a symbol sequence y, calculate the probability $P(w_j|y)$ for all words w_j in the L-th layer (i.e., the top layer). Store $P(w_j|y)$, and set $n = L$.
2. **Structural Search :** Select the m most probable words in the n-th layer, and calculate the probability $P(w_j|y)$ for their children in the $(n-1)$-th layer, where the probability calculation is omitted for the words that have already been examined. Store $P(w_j|y)$.
3. **Recognition :** Set n by $n - 1$, and repeat the step 2 until the bottom layer is reached (i.e., $n = 0$). Finally, select the most probable word as the recognition result from among the words having the probability.

The word candidates are effectively selected in each layer according to the word-node graph. Thus it is possible to restrict the search complexity, and to speed up the recognition

process. Furthermore, the lexical tree for scoring is constructed not for all words of the dictionary, but successively for the word candidates in each layer. For convenience, step 1 is called a initial search, the repetition of step 2 is called a structural search, and m in step 2 is called a beam width.

4.1. Computational Quantity

We estimate roughly the number of words to be examined. Assume that, for the dictionary of vocabulary size N, the word-node graph of L-layers is constructed as follows:

- The node number of $(n+1)$-th layer is half as many as that of n-th layer.
- Parent nodes in $(n+1)$-th layer are connected to 10 child-nodes in the n-th layer on the average.

Then the word number N_I for initial search is

$$N_I = \left(\frac{1}{2}\right)^L N, \tag{21}$$

and the word number N_S for structural search is

$$N_S = (10 \times m) \times L, \tag{22}$$

where m is beam width. For example, if $N = 2000, L = 3, m = 10$, then $N_I + N_S = 250 + 300 = 550$. The word duplication makes the total number of examined words less than $N_I + N_S$. Moreover, it is easily verified that if N_I and m are fixed, then N_S is restricted to $O(\log N)$. Thus, we see that our graph search is considerably effective for a large vocabulary word recognition.

5. Experimental Results

We compared the full search and our new search using a word-node graph. The 38 phoneme-based HMMs were used, where each HMM was a left-to-right model with no skipped states. The scoring was performed using the Viterbi algorithm. Our models were trained by 8 speakers' 1066 word utterances. As the test set, 4 different speakers' 1066 word utterances were used. The utterances were sampled at 12 kHz, and analyzed at 10-msec intervals. Table 1 shows the recognition accuracy and the number of words examined for the dictionary of 2029 words. The accuracy degraded according to the beam width, but the degradation was fairly small. With the beam width 15, the total number of examined words was restricted to 584 words. In this experiments, we used the normalized distance $\tilde{D}$ of (13) for the design of word-node graph. This graph had 3-layers. And the initial search had 257 words.

	full search	graph search		
beam width	—	20	15	10
word num.	2029	669	584	495
accuracy	93.8%	93.5%	93.2%	92.7%

Table 1 : Comparison between the full search and the graph search for the dictionary of 2029 words.

Similarly, Table 2 shows the results for the dictionary of 14993 words. We used a word-node graph of 5-layers. The initial search had 445 words. Comparing Table 1 and Table 2, we notice that the total number of examined words increased by only about 400, although the vocabulary size dramatically increased from 2029 to 14993.

	full search	graph search
beam width	—	16
word num.	14993	985
accuracy	90.9%	90.4%

Table 2 : Comparison between the full search and the graph search for the dictionary of 14993 words.

6. Conclusion

We have proposed a new search algorithm using a graph-structured dictionary. This is a word pre-selection technique. It is concluded that a distance measure defined by (9), the approximation of (4), is useful for word pre-selection. Moreover, the distance is calculated by the use of only model parameters. Therefore, we can easily design a word-node graph beforehand, based on the distance measure. Using this graph, we can significantly reduce the number of words to be examined, and obtain a recognition result at higher speed.

7. REFERENCES

1. L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proc. IEEE*, vol.77, no.2, pp.257–286, February 1989.
2. G. D. Forney, "The Viterbi algorithm", *Proc. IEEE*, vol.61, no.3, pp.268–278, March 1973.
3. T. Watanabe et al., "High Speed Recognition Using Tree-Structured Probability Density Function", *Proc. IEEE ICASSP*, vol.1, pp.556–559, 1995.
4. E. Bocchieri, "Vector Quantization for the Efficient Computation of Continuous Density Likelihood", *Proc. IEEE ICASSP*, vol.2, pp.692–695, 1993.
5. J.-K. Chen, F. K. Soong, and L.-S. Lee, "Large Vocabulary Word Recognition Based on Tree-Trellis Search", *Proc. IEEE ICASSP*, vol.2, pp.137–140, 1994.
6. B.-H. Juang and L. R. Rabiner, "A Probablistic Distance Measure for Hidden Markov Models", *AT&T Technical Journal*, vol.64, no.2, pp.391–408, February 1985
7. P. D'Orta, M. Ferretti, and S. Scarci, "Phoneme Classification for Real Time Speech Recognition of Italian", *Proc. IEEE ICASSP*, vol.1, pp.81–83, 1987.