Proposal of Hidden Structure Model *

Yu Qiao, Masayuki Suzuki, and Nobuaki Minematsu (The University of Tokyo)

1 Introduction

Speech recognition faces a challenge to deal with non-linguistic variations contained in speech signals. These variations are caused by the difference of speakers, communication channels, environment noise, etc. To overcome this difficulty, modern speech recognition approaches mainly make use of statistical methods (such as GMM and HMM) to model the distributions of acoustic features. These methods always require a large amount of data for training and can achieve relatively high recognition rates when there is a good match between training and testing data. But it is well-known that the performance of speech recognizers will drop significantly if there exists a mismatch. In our previous work [1], the third author proposed an invariant structural representation of speech which aims at removing the non-linguistic factors from speech signals. Different from classical speech models, the structural representations make use of contrastive features to model the global and dynamic aspects of speech and discard the local and static features. It can be proved that these contrastive features are invariant to any invertible transformations and thus are robust to non-linguistic variations [2]. We have already demonstrated the effectiveness of this representation in ASR [3, 4], speech synthesis [5], and CALL [6].

However, the structural representation also has its limitations. A speech structure is constructed for every sequence independently. So there may exist misalignment of events. Moreover, structural representations don’t have label information for events, which makes them difficult to use for general speech recognition. To overcome these difficulties, this paper proposes Hidden Structure Model (HSM) by introducing hidden states and probabilistic analysis. Compared with the previous structural representation, HSM unifies structure construction and structure comparison into a single framework, and avoids the misalignment of events. Moreover, the introduction of hidden states allows HSM to conduct structure-based decoding. This further allows us to apply HSM to general phoneme recognition other than word recognition. HSM is similar to HMM in a sense that both make use of hidden states, but different from HMM in a sense that HSM contains the probability models of both locally absolute and globally contrastive features. This paper proposes the fundamental formulation of HSM and develops the algorithms for state inference, probability calculation and parameter estimation. We carry out two experiments on artificial data and connected Japanese vowel utterances. The experimental results indicate that the combination of both absolute and contrastive features in HSM can improve the recognition rates.

2 Hidden Structure Model

In the previous structural representation, a distribution (event) sequence is calculated for each utterance independently of other utterances. There may exist misalignment between different distribution sequences. For example, let $P = \{p_1, p_2, \ldots\}$ and $Q = \{q_1, q_2, \ldots\}$ denote two distribution sequences calculated from two utterances of the same word ‘aiueo’. Assume that $p_3$ of $P$ comes from ‘i’, but $q_3$ of $Q$ may come from ‘u’. Another limitation of the structural representation is that it doesn’t include any label or category information of each event. Although the word recognition problem can be reduced to structure matching, it is difficult to extend this technique for other general speech recognition tasks, such as phoneme recognition. Moreover, in practice, contrastive features ($f$-divergences) are not strictly invariant due to noise and speaking styles. We may need to consider a probabilistic model for contrastive features.

We notice that HMM doesn’t have the above limitations. HMM avoids the misalignment problem by using DP-matching to align a cepstrum sequence with a sequence of HMM distributions. Moreover, HMM includes hidden states and has a flexible algorithm (Viterbi decoding) to estimate the most probable hidden state sequence for an observed sequence. This makes HMM suitable for solving general recognition tasks. Recall that the main advantage of the structural representation is that it makes use of contrastive features, which are robust to non-linguistic variations. Inspired by these facts, we develop Hidden Structure Model for sequence data, which aims at combining contrastive features with a flexible and probabilistic model. Like HMM, HSM introduces the hidden states of observations and takes account for the labels of these hidden states. Unlike HMM, HSM models the distributions of absolute and contrastive features, which makes it more robust to speaker differences.

2.1 Preprocessing of speech sequences for HSM

The contrastive features have to be calculated from events (sub-sequences or segments). For this reason, we need to divide a sequence $X = x_1, x_2, \ldots, x_M$ into a set of segments $O = o_1, o_2, \ldots, o_T$ in a preprocessing step (Fig. 1). Generally, we can use agglomerative clustering algorithm (ACA) [7] or HMM-based decomposition for sequence data [3, 4]. If we use ACA, each segment is a subsequence, denoted by $o_t = x_{m_1}, x_{m_1+1}, \ldots, x_{m_2}$, where $m_1$ to $m_2$ are the left and right borders of $o_t$. If we use the second method, each segment is modeled as a Gaussian distribution $N(\bar{o}_t, V_t)$. For each segment pair $o_t$ and $o_s$, we use $c_{ts}$ to denote the contrastive feature between them.
2.2 Introduction of Hidden Structure Model

Generally speaking, HSM is a probabilistic model for sequence data, which takes account for joint distribution of both absolute and contrastive features. To begin with, we formally describe the elements of HSM as the following.

1) $N$, the number of hidden states in HSM. We denote the set of individual states as $S = \{s_i\}_{i=1}^N$. We use $q_i$ ($q_i \in S$) to represent the hidden state of $o_i$ in sequence $O$. Then the state sequence is denoted by $Q = q_1, q_2, ..., q_T$.

2) State transition probability distribution $B = \{b_{ij}\}$, where $b_{ij} = p(q_{t+1} = s_j|q_t = s_i)$ $(1 \leq i, j \leq N)$.

3) Initial state distribution $\pi = \{\pi_i\}$, where $\pi_i = p(q_1 = s_i)$ $(1 \leq i \leq N)$.

4) Absolute observation probability (AOP) distribution $p(a_t|q_t = s_j)$ in state $j$. It is easy to see that the state inference problem is solved by Viterbi algorithm in the spirit of dynamic programming. However, it is

5) Contrastive observation probability (COP) distribution $p(c_{t_1,t_2}|q_{t_1} = s_i, q_{t_2} = s_j)$, where $c_{t_1,t_2}$ represents the contrastive features (BD, KL-div. [4, 2]) between $o_{t_1}$ and $o_{t_2}$. COP is assumed to have a Gaussian form, $p(c_{t_1,t_2}|q_{t_1} = s_i, q_{t_2} = s_j) = N(c_{t_1,t_2}|\mu_{i,j}^c, \Sigma_{i,j}^c)$. Let $A = \{\mu_{i,j}^c, \Sigma_{i,j}^c\}$ denote the set of COP parameters, and $O_A$ the set of absolute features of sequence $O$.

Fig. 2 An example of HSM. (HMM contains only the thick lines.)

and $O_C$, and the normalization factor reduces to 1. An example of HSM is depicted in Fig. 2. Note if we remove the contrastive part of Eq. 1, this probability calculation will be the same as that of HMM. On the other hand, if we remove the absolute part, Eq. 1 reduces to a probabilistic model of structural representation.

We introduce the following variables $Z = \{z_{i,t}\}$, where

$$z_{i,t} = \begin{cases} 1 & \text{if } q_t = s_i \\ 0 & \text{otherwise} \end{cases}$$

It is easy to see that $Z$ has the same information as $Q$. With $z_{i,t}$, we can rewrite Eq. 1 into

$$p(E|Z, \lambda) = \prod_{t=1}^{T} \prod_{i=1}^{N} p(a_t|s_i)^{z_{i,t}} \prod_{t=1}^{T-1} \prod_{i=1}^{N} \prod_{j=1}^{N} p(c_{t_1,t_2}|s_i, s_j)^{z_{i,t_1}z_{j,t_2}}.$$  

(2)

Like in HMM, the probability of state sequence is given by $p(Q|\lambda) = p(Z|\lambda) = p(Q|Z)p(Z|\lambda)$.

Log of the above equation is calculated as,

$$\log p(Q, Z|\lambda) = \sum_{t=1}^{T} \sum_{i=1}^{N} z_{i,t} \log \pi_i + \sum_{t=1}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} z_{i,t} z_{j,t-1} \log b_{i,j} + \sum_{t_1=1}^{T} \sum_{t_2=t_1+1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \sum_{l=1}^{N} \sum_{m=1}^{N} z_{i_1,t_1} z_{i_2,t_2} z_{j_1,t_1} z_{j_2,t_2}.$$  

(3)

where $z_{i,t} = \log p(a_t|s_i)$ and $\eta_{i,j,t_1,t_2} = \log p(c_{t_1,t_2}|s_i, s_j)$.

In the next, we introduce methods to solve the three problems of HSM, namely, state inference, probability calculation and parameter estimation.

2.3 State inference

Given model $\lambda$ and observed stream $O$, the objective of state inference is to determine $Z$ maximizing the following conditional probability, $\arg\max_Z p(Z|O, \lambda)$. Using Bayesian theory, we have

$$p(Z|O, \lambda) = \frac{p(O|Z, \lambda)}{p(O|\lambda)} \propto p(O|Z, \lambda).$$  

(4)

Thus the problem can be reduced to find $Z$ which maximizes Eq. 3, $\max_Z \log p(O, Z|\lambda)$. In HMM, the state inference problem is solved by Viterbi algorithm in the spirit of dynamic programming. However, it is
difficult to apply this technique to HSM. This is because in HSM we account for the full contrastive features between each two observations. In this paper, we make use of quadratic programming to solve Eq. 3. We can expand \( Z = \{z_{i,t}\} \) into an \( NT \)-dimensional vector \( z = [z_1, z_2, ..., z_T] \), where \( z_T = [z_{1,T}, z_{2,T}, ..., z_{N,T}] \). Similarly, we can introduce a vector \( d \) to represent the coefficients of first order parts of \( z \), and a matrix \( E \) to represent the coefficients of second order parts of \( z \).

Then the maximization of Eq. 3 can be written as the following 0-1 (binary) quadratic programming (QP) problem

\[
\max_z f(z) = zd^T + zEz^T, \tag{5}
\]

subject to: \( z_{i,t} \in \{0, 1\} \), \( \sum_t z_{i,t} = 1 \).

However, the above 0-1 QP is still very hard. To circumvent this difficulty, we relax the 0-1 constraint of \( z \) as the following one,

\[ 1 \geq z_{i,t} \geq 0, \quad \sum_i z_{i,t} = 1. \tag{6} \]

It can be proved that the relaxed QP has the same optimal solution as the original 0-1 QP of Eq. 5.

### 2.4 Probability calculation

In this section, we study the problem of how to calculate probability \( p(O|\lambda) \) of the observed sequence \( O \) given model \( \lambda \). Using marginal probability, we have

\[
p(O|\lambda) = \sum_Z p(O, Z|\lambda), \tag{7}
\]

To directly calculate the summations of the above equations is computationally expensive since there exist \( N^T \) possible paths of \( Q \). And HSM makes use of contrastive features, which prevent the usage of these DP-based algorithms for fast calculation. In this paper, we consider ‘a winner takes all’ (AWTA) approximation method. Let \( Z^* = \arg \max_Z p(O, Z|\lambda) \) denote the optimal solution of Eq. 3. Then we can approximate Eq.7 as

\[
p(O|\lambda) \approx \max_Z p(O, Z|\lambda) = p(O, Z^*|\lambda). \tag{8}
\]

Introduce expectation variables \( r_{i,t} = E[z_{i,t}] \) and \( \xi_{i,j,t_1,t_2} = E[z_{i,t_1}z_{j,t_2}] \). These can be calculated through summation, but this is time-costly. We can consider similar AWTA approximation as Eq. 8

\[
r_{i,t} \approx z^*_i; t\xi_{i,j,t_1,t_2} \approx z^*_i; t_1 z^*_j; t_2. \tag{9}
\]

### 2.5 Parameter estimation

In this section, we discuss the problem to estimate the parameters of HSM. Using maximum likelihood estimation (MLE), we have

\[
\arg \max_{\lambda} \prod_k p(O^k|\lambda), \tag{10}
\]

where \( O^k \) denotes the \( k \)-th training sequence. There doesn’t exist a closed form solution for MLE of HSM. So, we adopt EM algorithm [8] for optimization. Note \( \{r_{i,t}\} \) and \( \{\xi_{i,j,t_1,t_2}\} \) are the hidden parameters in EM iteration here.

In the E-step, given the old parameters \( \lambda^{\text{old}} \), we need to calculate the distribution of \( Z \) denoted by \( p(Z|O, \lambda^{\text{old}}) \). Since \( z_{i,t} \) is binary, this problem is reduced to estimate the expectations \( r_{i,t} \) and \( \xi_{i,j,t_1,t_2} \). There are two methods to do this. One is to estimate the marginal probabilities through summation. But this is computationally expensive. The other is to use the AWTA approximations. It is noted that these approximations are similar to the Viterbi training [9] of HMM (also known as segmental k-means), where the hidden parameters are determined through Viterbi alignment not by calculating marginal probabilities.

When the hidden parameters are given, we can find the model parameters maximizing the auxiliary function,

\[
Q(\lambda, \lambda^{\text{old}}) = \sum_k \sum_Z p(Z|O^k, \lambda^{\text{old}}) \log p(Z, O^k|\lambda). \tag{11}
\]

With hidden parameters \( r_{i,t} \) and \( \xi_{i,j,t_1,t_2} \), the optimal parameters \( \pi_i, \beta_{i,j}, \mu_i, \sigma_i, \mu_{i,j}, \Sigma_{i,j} \) can be estimated by maximizing \( Q(\lambda, \lambda^{\text{old}}) \).

### 3 Preliminary experiments

We carry out two preliminary experiments to examine HSM on labeling sequences. It is noted that the previous structural representation cannot conduct such a task.

#### 3.1 Experiment 1 with generated and transformed sequences

The first experiment examines the performance of HSM with artificially generated and transformed sequences. As preparation, we calculate the Gaussian distributions of cepstrum features for six symbols, i.e., five Japanese vowels (‘a’, ‘e’, ‘i’, ‘o’, ‘u’) and silence (‘sl’). Using the six symbols, we randomly generate a set of strings, the length of each of which is fixed to 16. Then the corresponding cepstrum features of these strings are obtained by using the Gaussian models to generate five frame vectors for each symbol. After that, we perform acoustic transformation on the generated cepstrum features as if the features are generated by different speakers. The acoustic transformation is realized by frequency warping, which corresponds to multiplication of a specific type of matrix by cepstrum vectors [10]. The elements of the matrix are functions of warping parameter \( \alpha \) [10] and, by changing the value of \( \alpha \), we can lengthen/shorten the vocal tract length of speakers. Here, the value of \( \alpha \) with a uniform distribution in [-0.5,0.5] is randomly selected. In this way, we generate a set of strings which are acoustically realized by different speakers. Using this procedure, a set of transformed cepstrum sequences are prepared for training and another set for testing. It should be noted that the sequences are different strings and are acoustically realized by different speakers.

We train a single six-state HSM from the sequences in the training set. It is noted that since label (symbol) and boundary information of every sequence is known, we can directly estimate the parameters of the distributions of the absolute and contractive features without EM.
iterations. Once the HSM is trained, we use the QP-based state inference method developed in Section 2.3 to estimate the symbol (state) information of every testing cepstrum sequence. In other words, each input sequence is aligned to the HSM. The testing set contains 20 sequences. We change the number of training sequences from 10 to 40. For each case, we repeat the experiments 20 times. The average symbol-based recognition rates are shown in Fig. 3, where ‘Ab’ represents the use of absolute features only, and ‘Ab+Cn’ the use of both absolute and contrastive features. ‘Original’ means using the original Gaussian distributions for training. As discussed in the beginning of Section 2, the absolute feature only is essentially the same as HMM. As one can see, the combination of both absolute and contrastive features has the best performance.

3.2 Experiment 2 with Japanese vowel utterances

We also examine the performance of HSM for labeling sequences with a database of continuously connected Japanese vowel utterances. It is known that acoustic features of vowel sounds exhibit larger between-speaker variations than consonant sounds. Each word in the data set is a concatenation of the five Japanese vowels ‘a’, ‘e’, ‘i’, ‘o’ and ‘u’, such as ‘aeiou’, ‘uoaie’, etc. So there are totally 120 words. The utterances of 16 speakers (8 males and 8 females) were recorded. Every speaker provides 5 utterances for each word. The total number of utterances is 16×120×5=9,600. For each utterance, we calculate twelve Mel-cepstrum features and one power coefficient. Then ML-based decomposition is used to convert cepstrum vectors as a sequence of 25 Gaussian distributions (sub-segments) [4, 3]. We label each distribution as ‘a’, ‘e’, ‘i’, ‘o’, ‘u’ or ‘sl’ by forced alignment with speaker-dependent phoneme HMMs.

We train independently a 6-state HSM with 600 (120×5) utterances of a male speaker and another HSM for a female speaker. For each HSM, we examine its state recognition rates for utterances of other male speakers, other female speakers, and both. The results are summarized in Fig. 4. We found that ‘Ab+Cn’ achieves the best recognition rates. Moreover, the contrastive features have better performance than the absolute features.

4 Conclusions

This paper proposes Hidden Structure Model (HSM) for sequence data. HSM generalizes our previous structural representation into a probabilistic framework, which accounts for both absolute and contrastive features. Like HMM, HSM makes use of hidden states. Different from HMM, HSM contains the distributions of contrastive features. We also develop algorithms for state inference, probability calculation, and parameter estimation of HSM. Due to the usage of contrastive features, we cannot use dynamic programming to develop HMM-like algorithms, such as Viterbi algorithm, forward and backward algorithm, and Baum-Welch algorithm. In this paper, we formulate the state inference as a quadratic programming problem, and develop approximation methods for probability calculation and parameter estimation. We conducted two preliminary experiments to examine the performance of HSMs. One is on artificially generated sequences, the other makes use of connected Japanese vowel utterances. The results show the usefulness of HSM and advantages of combining absolute and contrastive features. The experiments of this paper are basic and limited. We are going to improve HSM, and apply HSM to other tasks.

參考文獻