State-level variable modeling for phoneme classification

Hao-Zheng Li, Douglas O'Shaughnessy

INRS-EMT University of Quebec
{lihz,dougo}@emt.inrs.ca

Abstract

In HMM-based pattern recognition, the structure of the HMM is often predetermined according to some prior knowledge. In the recognition process, we usually make our judgment based on the maximum likelihood of the HMM, without considering the time-varying property of state-level variables, which unfortunately may lead to incorrect results. In this paper, we analyze the property of state-level variables in the HMM and show it is possible to significantly enhance the performance of speech recognition systems when using the state-level variable time-varying property. We propose four methods to model state-level variable trajectories and then test them on a phoneme classification task on the TIMIT speech corpus, 11.95% error rate reduction is achieved and some empirical conclusions are drawn.

Index Terms: speech recognition, Hidden Markov Model, state-level variables.

10.21437/Interspeech.2006-220

1. Introduction

HMM (Hidden Markov Model) theory, especially on how to learn the topology of HMM, is far from completed. Hidden states are used to encode the surrounding context and it is essential to choose an appropriate topology for an accurate HMM [2]. However, in most applications, the structure of an HMM is selected based on prior knowledge and the individual hidden state variables are associated with different observation segments. For example, for isolated word recognition with a distinct HMM designed for a word in the vocabulary, the number of states corresponds roughly to the number of sub-word units (e.g., phoneme) within the word and the structure is left-to-right [1]. This implies that a particular linguistic segment (phoneme) is forced to be attached to each state that corresponds to a certain segment of the observation sequence.

This paper will not focus on the learning of the topology but on how to incorporate the additional information because of this forcing. The state-level variable at different segments should reflect the nature of the segment the state represents, i.e., given an observation sequence, the state in a model should have a higher posterior probability value for the component it represents than other states for the same component. However, the HMM-based recognizer does not take the change of state-level variable into consideration. In this paper, we will analyze the characteristics of state-level variables and show that using the state-level information in the HMM will be able to improve recognition results.

In the following section we briefly review the basics of the hidden Markov model. In Section 3 we analyze the characteristics of state-level variables in an HMM and possible ways to use state-level information for recognition. We show our experimental results in Section 4, which is followed by a conclusion in Section 5.

2. Hidden Markov Models

In an HMM with N underlying states, a state sequence $S = (S_1, S_2, \cdots, S_T)$ generated by the Markov chain cannot be directly observed, but only through the observation sequence $Y = (Y_1, Y_2, \cdots, Y_T)$ result from the state sequence according to the observation distribution defined by $B = \{b_i(Y_t) : 1 \leq i \leq N\}$, with $b_i(Y_t) = P(Y_t|S_t = i)$. The transition from state i to state j is specified by an $N \times N$ matrix $A = [A_{ij}]$ with $A_{ij} = P(S_t = j|S_{t-1} = i)$. $\pi = [\pi_1, \pi_2, \cdots, \pi_N]$ is the initial state probability vector with $\pi_i = P(S_1 = i)$. λ is the compact notation for the model parameters in an HMM. There are three key problems of interest that must be solved for the model to be useful in real-world applications. These problems are the following.

A. The Classification Problem

The probability of observation sequence Y given the λ, $P(Y|\lambda)$, can be used to perform classification. The computation of $P(Y|\lambda)$ is significantly simplified when defining the forward variable and backward variables [3]:

$$\alpha_t(j) = P(Y_1, \cdots, Y_t, S_t = j|\lambda), \quad (1)$$

which is the probability of observing the first t observation vectors and visiting the j th state at time t. Similarly, the backward variable is

$$\beta_t(j) = P(Y_{t+1}, \cdots, Y_T|S_t = j, \lambda). \quad (2)$$

At any t, $P(Y|\lambda) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j)$. If the HMMs are accurate enough, the $P(Y|\lambda)$ would be exactly the $P(Y)$ and does not need any additional information. However, because of the forcing of the topology of the HMMs or other

reasons, using an additional heuristic information is possible.

B. The Optimal-State Sequence Problem

This is the computation of the state sequence that fits best to an observed sequence. The Viterbi algorithm can solve this problem. In the Viterbi algorithm a quantity delta is defined:

$$\delta_t(j) = \max_{S_1,S_2,\cdots,S_{t-1}} P\left(S_1, S_2, \cdots, S_t = j, Y_1^t|\lambda\right), \quad (3)$$

which is the highest likelihood of observing the first t observations along a single path, $S_1, S_2, \cdots, S_t$ with $S_t = j$ and can be computed inductively. A backtracking procedure can be used to obtain the optimal-state sequence.

C. The Training Problem

This is the computation of the model parameters A, B and π to maximize the probability of observation sequences in the training set. The Baum-Welch algorithm is often used to train parameters which heavily depend on the a posteriori probability variable

$$\gamma_t(i) = \alpha_t(i)\,\beta_t(i)\,/P(Y|\lambda), \quad (4)$$

which is the probability of being in state i at time t, given the observation sequence Y and the model λ.

3. State-Level variables

In [1], Bengio has shown that a left-to-right model is more appropriate than an ergodic model in speech recognition. Fig. 1 is the most commonly used HMM structure in phoneme-based ASR. In the rest of this paper, all the investigations are based on Fig. 1. Consider only the numerator in Equation

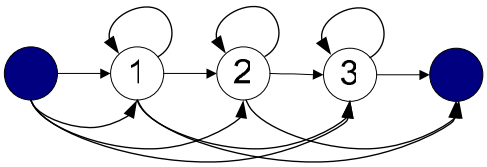


Figure 1: Left-To-Right HMM graph

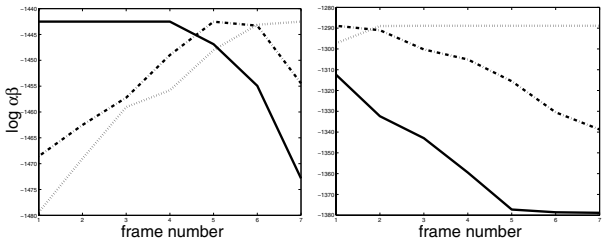
(4) , which reflects the change of posteriori state probability for state i at different times t. If we link them from $t = 1$ to $t = T$ and thus form a gamma trajectory for state i. Gamma trajectories of different state will have different evolutions and reflect the role of the hidden state.

Fig. 2(a) shows the gamma trajectories of three states for a phoneme 'aa' from the TIMIT database, where the solid line is the trajectory of state 1, the dashed-dot line represents the second trajectory, and the dotted line represents the third trajectory. Fig. 2(a) conveys at least two phenomena:

- Each state has the highest a posteriori probability at its corresponding local segment. Beyond this local segment, the state a posteriori probability declines rapidly. That is, a state can only model well its corresponding local segment.
- For a given segment, a state may have a higher a posteriori probability value than other states if it is closer to the state that represents the segment. For example, at segment 1 that is modeled by state 1, state 2 will have a higher a posteriori probability value than state 3, since state 2 is closer to state 1 than state 3.

We call the trajectories like in Fig. 2(a) as the typical trajectories because they reflect the role of hidden states.

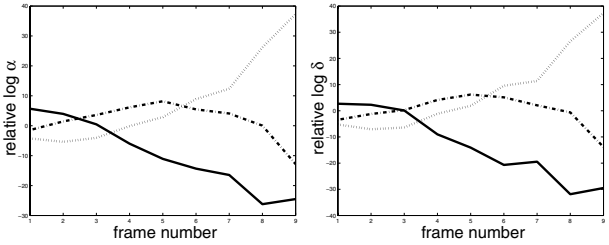
By comparison, Fig. 2(b) shows atypical log gamma trajectories of the three states for one of the phonemes labeled as 'iy' that has been mistaken as 'aa' by the conventional HMM-based recognizer.



(a) Typical (b) Atypical

Figure 2: Log gamma trajectories of three states

In fact the α variable and the δ variable (δ can be seen as an approximation of the α variable) have similar time evolutions as the γ variable. Fig. 3(a) shows the relative log alpha trajectories of three states (to display conveniently, at every t, $log(\alpha_t(j))$ is subtracted from the mean of $log(\alpha_t(j))$ $(j = 1, 2, 3)$ and we call them relative log alpha trajectories). Fig. 3(b) shows the relative log delta trajectories of three states. The two trajectories are very similar; they both convey the same two phenomena as Fig. 2(a).



(a) alpha (b) delta

Figure 3: Relative Log alpha and delta trajectories of three states

Gamma, alpha, delta state-level trajectories have similar time evolutions, which will reflect the role of hidden states.

The rest of this paper will only use the gamma trajectories as the example.

4. Modeling State-Level trajectories

State-level trajectories are a type of coupled sequential data; how to model them is a major problem. Real speech signals of the same speech unit may vary significantly and gamma trajectories also show great diversity. We can seek some properties of an ensemble for state-level trajectories. In this paper, we will use the following four methods to capture the major properties of the trajectories. Each following method is a refinement of its preceding method.

A. Computing the number of intersecting points (NIPs)

First, we use a simple method to capture the major properties of the trajectories of different states by computing the total number of intersecting points (NIPs) of each pair of trajectories. For HMMs with N states, the typical NIPs is C_N^2, which means each pair of trajectories has only one intersecting points. For the HMMs in Fig. 1, Fig. 4 shows the histogram of NIPs of the gamma trajectories for the phonemes 'aa' (solid line) and for the phonemes mistaken for 'aa' (dotted line) in TIMIT training set by a conventional recognizer. We can see from Fig. 4 that the two histograms are quite different, especially when the NIPs is less than 4.

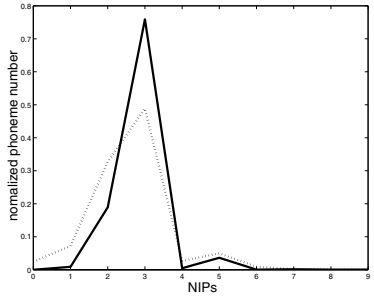


Figure 4: Histogram of NIPs of three gamma trajectories

For HMMs with N states, only one heuristic NIPs density is needed.

B. Computing the number of separate intersecting points (NSIPs)

Although the NIPs may provide some discriminative information for HMMs, it still makes some confusions in some cases. For example, for N=3, if a pair of gamma trajectories have 3 intersection points while other pairs don't have any, they are the atypical gamma trajectories. To reduce this kind of confusion, we propose to compute the number of separate intersecting points for each pair of trajectories respectively.

For HMMs with N states, C_N^2 heuristic NSIPs densities are needed.

C. Computing the number of separate discriminative intersecting points (NSDIPs)

By further refining the above method, we classify the intersection points on each pair of trajectories into two types. Type one is: on the left of an intersecting point, $\{^{ind(i)<ind(j)}_{\gamma_t(i)>\gamma_t(j)}$, on the right,$\{^{ind(i)<ind(j)}_{\gamma_{t+1}(i)<\gamma_{t+1}(j)}$, where $ind(i)$ denotes the index number of the state i; type two is: on the left of an intersecting point, $\{^{ind(i)>ind(j)}_{\gamma_t(i)>\gamma_t(j)}$, on the right,$\{^{ind(i)>ind(j)}_{\gamma_{t+1}(i)<\gamma_{t+1}(j)}$

We call this method as computing the number of separate discriminative intersecting points (NSDIPs). For HMMs with N states, $2 * C_N^2$ heuristic NSIPs densities are needed.

D. Computing the position of separate discriminative intersecting points (PSDIPs)

Computing the position of separate discriminative intersecting points would further refine the above methods. Here we propose a novel method, which can be illustrated by Fig. 5, to keep track of the position of only some key intersecting points.

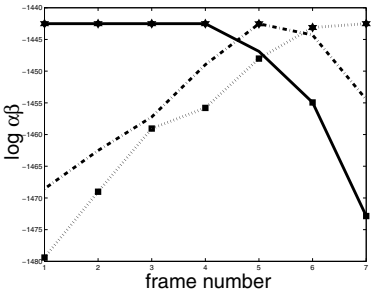


Figure 5: Optimal and worst state sequence on gamma trajectories

From Fig. 5, we may obtain the optimal state sequence $(s_1, s_1, s_1, s_1, s_2, s_3, s_3)$(marked with stars) and the worst state sequence $(s_3, s_3, s_3, s_3, s_3, s_1, s_1)$(marked with squares). From the two kinds of state sequences, we know there exist three intersecting points: point between $t = 4$ and $t = 5$ for $\gamma_t(1)$ and $\gamma_t(2)$; $t = 5$ and $t = 6$ for $\gamma_t(2)$ and $\gamma_t(3)$; $t = 5$ and $t = 6$ for $\gamma_t(1)$ and $\gamma_t(3)$.

So we will keep the heuristic optimal state sequence and worst state sequence information, which is used to keep track of the position of separate discriminative intersecting points (PSDIPs). For HMMs with N states, $2 * N$ heuristic PSDIPs densities are needed.

To use the addtional information provided by the state-level trajectories, we modify the observation probability $P(Y|\lambda)$ as follows:

$$\log \tilde{P}(Y|\lambda) = \log P(Y|\lambda) + \sigma \sum_{j=1}^{Q} \log p(d_j), \quad (5)$$

where σ is a scaling multiplier, Q is the number of heuristic densities and $p(d_j)$ denotes the probability computed from the jth densities.

5. Experimental Results

We have carried out several experiments to test our model on a speech phoneme classification task, i.e., assuming the segmentation time is known, the task will recognize the unigram and context-independent phonemes. We used the phonetically balanced TIMIT speech database. Our training and testing sets are created with the 'sx' and 'si' training and testing sentences from TIMIT with 3696 and 1344 distinct sentences, respectively. The standard phonetic clustering [4] was used, resulting in 39 phoneme models. We use HTK [5] as the baseline, in which the three-state left-to-right models as shown in Fig. 1 are adopted.

We compared the performances of five methods: (1) HMM only; the other four make use of state-level information: (2) HMM with the NIPs for the gamma, alpha, delta trajectories; (3) HMM with the NSIPs for the three kinds of trajectories; (4) HMM with the NSDIPs for the three kinds of trajectories; (5) HMM with the PSDIPs for the three kinds of trajectories. Gaussian mixtures were used for the output probabilities. To test whether the proposed methods perform better on *good* acoustic models or on *poor* acoustic models, we use 8 mixture components per state as the *poor* model and 64 mixture components per state as the *good* model. 24 mixture components per state is used as the *average* model. We used 12 Mel Cepstral coefficients, plus the energy parameter, and their first and second order difference as the output feature. Thus the total dimension of the feature vector is 39. These parameters are derived from 32 ms long window frames with a 22 ms overlap in each of the two adjacent frames. In all methods, computing probability about the intersection points adds little in terms of the total computational cost, since the computational cost is mostly caused by computing the observation probability required by a conventional HMM-based recognizer. All methods concerning the delta trajectories use the Viterbi as the decoding method. We use the histogram for computing probability $p(k)$. The recognition rates are given in Table 1. We can see from Table 1 that in all methods gamma trajectories are better than alpha trajectories; this is as expected because the gamma variables are determined by the past and future context while the alpha variables are determined by the past context. Also, in all cases alpha trajectories are better than delta trajectories, since delta can be seen as an approximation of alpha.

From NIPs to PSDIPs, the intersection points are gradually better modeled, and the improvements become more and more. The best result with 8 mixtures (67.88%) causes 8.78% error rate reduction, while with 24 mixtures cause an 11.26% reduction and with 64 mixtures result in an 11.95% reduction. This means state-level variable will reflect the role of states better in a good acoustic model, and thus bring more improvements.

Table 1: Comparison of HMMs with different methods and different trajectories.

method	trajectory	8 mix	24 mix	64 mix
baseline		64.79	68.47	71.13
NIPs	gamma	65.92	70.22	72.99
	alpha	65.88	70.11	72.98
	delta	65.85	70.05	72.88
NSIPs	gamma	66.22	70.82	73.49
	alpha	66.00	70.55	73.33
	delta	65.91	70.45	73.27
NSDIPs	gamma	66.27	70.99	73.62
	alpha	66.21	70.88	73.53
	delta	66.12	70.84	73.46
PSDIPs	gamma	67.88	72.02	74.58
	alpha	67.70	71.93	74.48
	delta	67.65	71.90	74.43

6. Conclusions and Future Work

In this paper, we have shown that the state-level variables have their own specific time-varying properties, which can reflect the roles of individual hidden states of the HMM. We have proposed four methods for using this type of state-level information in recognition tasks. From our initial investigation we have found that the more accurately we model the intersection points, the better improvements will be achieved. We also found the better the acoustic model is, the larger error rate reduction can be obtained. So far, we model the intersecting points of the trajectories only; this probably can be significantly improved by modeling the state-level variables trajectories directly.

7. References

[1] Y. Bengio, "Markovian models for sequential data," *Neural Computing Surveys*, vol. 2, pp. 129–162, 1999.

[2] J. Bilmes, "What HMMs Can Do," *UWEE Technical Report Number UWEETR-2002-0003*, *https://www.ee.washington.edu/techsite/papers/documents/UWEETR-2002-0003.pdf* , 2002.

[3] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, pp. 257–286, 1989.

[4] K. Lee and H. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Tran. ASSP*, vol. 37, pp. 1641–1648, 1989.

[5] S. Young, et. al. "The HTK Book for HTK V3.3," *http://htk.eng.cam.ac.uk*, 2005.