Journal of Southeast University (English Edition) Vol. 21, No. 2, pp. 136 - 140 June 2005 ISSN 1003—7985

Optimal state and branch sequence based parameter estimation of continuous hidden Markov model

Yu Lu[1,2] Wu Lenan[1] Xie Jun[3]

([1] Department of Radio Engineering, Southeast University, Nanjing 210096, China)

([2] Institute of Communications Engineering, PLA University of Science and Technology, Nanjing 210007, China)

([3] Institute of Command Automation, PLA University of Science and Technology, Nanjing 210007, China)

Abstract: A parameter estimation algorithm of the continuous hidden Markov model is introduced and the rigorous proof of its convergence is also included. The algorithm uses the Viterbi algorithm instead of K-means clustering used in the segmental K-means algorithm to determine optimal state and branch sequences. Based on the optimal sequence, parameters are estimated with maximum-likelihood as objective functions. Comparisons with the traditional Baum-Welch and segmental K-means algorithms on various aspects, such as optimal objectives and fundamentals, are made. All three algorithms are applied to face recognition. Results indicate that the proposed algorithm can reduce training time with comparable recognition rate and it is least sensitive to the training set. So its average performance exceeds the other two.

Key words: continuous hidden Markov model; optimal state and branch sequence; maximum likelihood; convergence; Viterbi algorithm

The hidden Markov model (HMM) is a statistical model to depict nonstationary stochastic processes. It has widespread applications in all areas of pattern recognition. Parameter estimation has been an active and significant field in the study of the HMM all along. The traditional Baum-Welch (denoted by BW henceforth) algorithm[1] and the segmental K-means (denoted by KV) algorithm[2] use maximum-likelihood (ML) as optimization criterion. Although some other optimal objectives[3,4] were introduced later, the training algorithms based on ML are still the most popular.

What we discuss in the paper are training algorithms based on ML. The BW algorithm adds the probabilities over all possible state transition paths during estimation; the KV algorithm (also named as the training algorithm grounded on optimal state path) finds the optimal state sequence by the Viterbi algorithm and determines the branch sequence using K-means clustering, while our algorithm determines optimal (in ML sense) state and branch sequences via the Viterbi algorithm directly.

1 Fundamentals of the HMM

The parameter set to characterize a continuous HMM includes initial distribution $\boldsymbol{\pi} = [\pi_i]$, transition matrix $\boldsymbol{A} = [a_{ij}]$ and emit probability density function (PDF) $b_j(\boldsymbol{O}_t)$, $\boldsymbol{O}_t$ is the observation vector at time t.

$$\pi_i = P(q_1 = i) \quad 1 \leqslant i \leqslant N, \sum_{i=1}^{N} \pi_i = 1$$
$$a_{ij} = P(q_{t+1} = j \mid q_t = i) \quad 1 \leqslant i,j \leqslant N; \sum_{j=1}^{N} a_{ij} = 1 \tag{1}$$

where N is the number of states of the model, q_t denotes the state at time t. Usually the emit PDF is approximated by a weighted sum of M Gaussian density functions G.

$$b_j(\boldsymbol{O}_t) = \sum_{k=1}^{M} b_{jk}(\boldsymbol{O}_t) = \sum_{k=1}^{M} c_{jk} G(\boldsymbol{O}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$$
$$c_{jk} \geqslant 0, 1 \leqslant j \leqslant N, 1 \leqslant k \leqslant M, \sum_{k=1}^{M} c_{jk} = 1 \tag{2}$$

where M is the number of mixture components which are called "branches" in this paper, $\boldsymbol{\mu}_{jk}$ is the mean vector and $\boldsymbol{\Sigma}_{jk}$ is the covariance matrix of the k-th branch in state j. Let $\boldsymbol{C} = [c_{jk}]$, $\boldsymbol{\mu} = [\boldsymbol{\mu}_{jk}]$, $\boldsymbol{\Sigma} = [\boldsymbol{\Sigma}_{jk}]$, then a continuous HMM can be characterized by $\lambda = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{C}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$.

For the mixture model formulated by Eq. (2), Ref. [5] made the following illustration which is shown in Fig. 1. A state j with M branches is equivalent to $M+2$ states $j, j_0, j_1, j_2, \ldots, j_M$, each with a single branch. The transitions departing from state j have probabilities equal to the corresponding weights c_{jk}, $1 \leqslant k \leqslant M$, and each state j_k, $1 \leqslant k \leqslant M$ generates observation $\boldsymbol{O}$ according to the probability distribution $G(\boldsymbol{O}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$, $1 \leqslant k \leqslant M$. At state j and j_0 no observation is generated. That is, the external behaviour of a

Received 2004-12-21.

Biographies: Yu Lu (1973—), female, graduate; Wu Lenan (corresponding author), male, doctor, professor, wuln@seu.edu.cn.

state with M branches is to generate $G(\boldsymbol{O}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$, $1 \leqslant k \leqslant M$ *distributed observations with probability* c_{jk}, $1 \leqslant k \leqslant M$. An analogous illustration can also be found in Ref. [6].

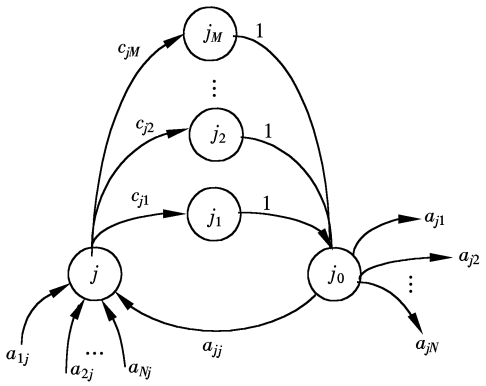

Fig. 1 Illustration in Ref. [5] to the mixture model

2 Parameter Estimation of Continuous HMM Based on Optimal Sequence

The mechanism of generating observations in the HMM shows that both the choice of branch and the choice of the next state are in terms of some discrete distribution. This is why we can design a training algorithm based on optimal state and branch sequences. Before formulation of the algorithm, its relation to the BW and KV algorithms will be presented.

2.1 Relationship with the other two algorithms

Our algorithm, the BW algorithm and the KV algorithm are all optimal procedures based on maximum-likelihood but differ in respective optimal objectives. For the training sequence $\boldsymbol{O}$, the BW algorithm aims at a maximum of $P(\boldsymbol{O} \mid \lambda)$. In all the probability formulas of this paper, the occurrence of observation sequence $\boldsymbol{O}$ means that the observation sequence falls into a little neighbor field of $\boldsymbol{O}$ (Because in a continuous distribution, the probability that the observation sequence equals a specified value is equal to 0). And in all the formulas, a common factor (general volume of Δo) is omitted which will not change the ratio of probabilities. The KV algorithm tries to maximize $\max\limits_{\boldsymbol{Q}} P(\boldsymbol{O}, \boldsymbol{Q} \mid \lambda)$ ($\boldsymbol{Q}$ is the state sequence), while our algorithm wants to find $\max\limits_{\boldsymbol{Q}, \boldsymbol{K}} P(\boldsymbol{O}, \boldsymbol{Q}, \boldsymbol{K} \mid \lambda)$ ($\boldsymbol{K}$ is the branch sequence). As to the optimal objective, the relation between our algorithm and the KV algorithm is the same as the relation between the KV algorithm and the BW algorithm, that is, the connection between maximum and sum. In many applications, the maximum is a good approximation to the sum and the difference between them is very small if only the observation sequence is long enough.

It should be pointed out that although the KV algorithm uses $\max\limits_{\boldsymbol{Q}} P(\boldsymbol{O}, \boldsymbol{Q} \mid \lambda)$ as its objective function, it determines the branch sequence by K-means clustering instead of summing the probabilities over all possible branch sequences in practical implementation. The difference between our algorithm and the KV algorithm lies in the way to determine what is the most likely state and branch sequence. In our algorithm, the Viterbi algorithm is used to determine optimal state and branch sequences directly. Not using K-means clustering, which is sensitive to the initial cluster, the determination of the optimal branch sequence is more creditable in theory and takes less time.

2.2 Determination of optimal state and branch sequence

Viterbi is the classical algorithm to determine optimal path in dynamic programming. In our algorithm, the Viterbi algorithm is used to determine optimal state and branch sequences. To do this, an auxiliary function is defined.

$$\delta_t(j,k) = \max_{\substack{q_1,q_2,\ldots,q_{t-1} \\ k_1,k_2,\ldots,k_{t-1}}} P(q_1, q_2, \ldots, q_{t-1}, k_1, k_2, \ldots, k_{t-1}, q_t = j, k_t = k, \boldsymbol{O}_1, \boldsymbol{O}_2, \ldots, \boldsymbol{O}_t \mid \lambda) \qquad t > 1 \tag{3}$$

$$\delta_1(j,k) = P(q_1 = j, k_1 = k, \boldsymbol{O}_1 \mid \lambda) = \pi_j c_{jk} G(\boldsymbol{O}_1, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) \tag{4}$$

Then the following recursive relationship holds:

$$\delta_{t+1}(j,k) = \max_{\substack{1 \leqslant i \leqslant N \\ 1 \leqslant l \leqslant M}} [\delta_t(i,l)\, a_{ij}]\, c_{jk} G(\boldsymbol{O}_{t+1}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) \tag{5}$$

and the joint probability for optimal state and branch sequence and observation sequence $\boldsymbol{O}$ can be derived:

$$\max_{\boldsymbol{Q}, \boldsymbol{K}} P(\boldsymbol{O}, \boldsymbol{Q}, \boldsymbol{K} \mid \lambda) = \max_{\substack{1 \leqslant j \leqslant N \\ 1 \leqslant k \leqslant M}} \delta_L(j,k) \tag{6}$$

where L is the length of the observation sequence.

2.3 Parameter estimation based on optimal state and branch sequence

Similar to the two classical algorithms, our algorithm achieves optimization also by iteration. There are two steps in iteration at time t: the first is to compute

$$(\boldsymbol{Q}^*(t-1), \boldsymbol{K}^*(t-1)) = \arg \max_{\boldsymbol{Q}, \boldsymbol{K}} P(\boldsymbol{O}, \boldsymbol{Q}, \boldsymbol{K} \mid \lambda(t-1)) \tag{7}$$

where $\lambda(t-1)$ is already known. The second is to determine

$$\lambda(t) = \arg \max_{\lambda} P(\boldsymbol{O}, \boldsymbol{Q}^*(t-1), \boldsymbol{K}^*(t-1) \mid \lambda) \tag{8}$$

where $(\boldsymbol{Q}^*(t-1), \boldsymbol{K}^*(t-1))$ has already been obtained in the first step.

To accomplish step one, we can resort to the recursion (5) and backtracking. Now, let us turn to estimation of parameters of the HMM with optimal state and branch sequence already known.

For the sake of convenience, we throw off the time mark, and let $\boldsymbol{Q}^*(t-1) = \boldsymbol{Q}^* = \{q_1, q_2, \ldots, q_L\}$,

$\boldsymbol{K}^*(t-1)=\boldsymbol{K}^*=\{k_1,k_2,\ldots,k_L\}$. By the monotonicity of logarithms, the optimization of log-likelihood is equivalent to the optimization of likelihood. Then

$$\begin{aligned}&\log P(\boldsymbol{O},\boldsymbol{Q}^*(t-1),\boldsymbol{K}^*(t-1)\mid\lambda)=\\&\quad\log P(\boldsymbol{Q}^*\mid\lambda)+\log P(\boldsymbol{K}^*\mid\boldsymbol{Q}^*,\lambda)+\\&\quad\log P(\boldsymbol{O}\mid\boldsymbol{Q}^*,\boldsymbol{K}^*,\lambda)=\\&\quad\sum_{i=1}^{N}\sum_{j=1}^{N}N_{ij}\log a_{ij}+\sum_{i=1}^{N}M_i\log\pi_i+\\&\quad\sum_{j=1}^{N}\sum_{k=1}^{M}L_{jk}\log c_{jk}+\sum_{j=1}^{N}\sum_{k=1}^{M}b_{jk}(\boldsymbol{O}_t)\end{aligned}\tag{9}$$

where N_{ij} is the times of occurrence of $q_t=i, q_{t+1}=j$ in optimal state sequence from $t=1$ to $L-1$, and M_i is the times of occurrence of $q_1=i$. In fact, M_i is equal to 0 or 1. Similarly, L_{jk} is the occurrence times of $q_t=j, k_t=k$ in optimal state and branch sequence from $t=1$ to L. And $\boldsymbol{O}_t$ is the observation at t which satisfies $q_t=j, k_t=k(1\leqslant t\leqslant L)$, namely, $\boldsymbol{O}_t$ is the observation generated by branch k of state j.

The items in Eq. (9) depend on different parameters so they can be optimized separately. Optimizing over the first three items with the constraint condition (1) and (2), we can get

$$\begin{aligned}\bar{a}_{ij}&=\frac{N_{ij}}{\sum_{j=1}^{N}N_{ij}},\quad\bar{\pi}_i=\frac{M_i}{\sum_{j=1}^{N}M_j}\qquad 1\leqslant i,j\leqslant N\\\bar{c}_{jk}&=\frac{L_{jk}}{\sum_{l=1}^{M}L_{jl}}\qquad 1\leqslant k\leqslant M\end{aligned}\tag{10}$$

The last item in Eq. (9) is the sum of several independent log-likelihood. They can also be optimized separately.

For the branch k of state j, supposing observations $\boldsymbol{x}_{jk}^{(1)},\boldsymbol{x}_{jk}^{(2)},\ldots,\boldsymbol{x}_{jk}^{(l)}$ are generated by the maximum-likelihood estimation of multivariate normal distribution, we get

$$\begin{aligned}\bar{\boldsymbol{\mu}}_{jk}&=\frac{1}{l}\sum_{p=1}^{l}\boldsymbol{x}_{jk}^{(p)}\\\bar{\boldsymbol{\Sigma}}_{jk}&=\frac{1}{l}\sum_{p=1}^{l}(\boldsymbol{x}_{jk}^{(p)}-\bar{\boldsymbol{\mu}}_{jk})(\boldsymbol{x}_{jk}^{(p)}-\bar{\boldsymbol{\mu}}_{jk})^{\mathrm{T}}\end{aligned}\tag{11}$$

The estimated parameters in Eqs. (10) and (11) are similar to the results of the KV algorithm because both are estimations based on some certain state and branch sequence.

2.4 Proof of convergence

The Zangwill global convergence theorem, which is the most important theoretic tool in analysis of algorithm's convergence, will be used to prove the convergence of the algorithm in the paper.

A continuous HMM $\lambda=\{\boldsymbol{\pi},\boldsymbol{A},\boldsymbol{C},\boldsymbol{\mu},\boldsymbol{\Sigma}\}$ can be viewed as a point in the space $\mathbf{R}^p$ (Where p is the number of parameters to model a continuous HMM). All valid models which satisfy the constraint condition (1) and (2) form a closed subset of $\mathbf{R}^p$, named as Λ. Usually, only HMMs with bounded parameters make sense, so the boundedness of Λ is often assumed[2,7]. With this assumption, Λ is compact because it is a bounded and closed set in $\mathbf{R}^p$.

The algorithm in the paper can be written as a compound mapping.

$$\lambda(t-1)\xrightarrow{T_1}(\boldsymbol{Q}^*(t-1),\boldsymbol{K}^*(t-1))\xrightarrow{T_2}\lambda(t)\tag{12}$$

$(\boldsymbol{Q}^*(t-1),\boldsymbol{K}^*(t-1))$ is defined in Eq. (7) and $\lambda(t)$ is defined in Eq. (8). Assume S is the set of all the possible state and branch sequences. It is easy to see that $T_1:\Lambda\to S$ is point to set and $T_2:S\to\Lambda$ is point to point, while the compound mapping is point to set.

$$T:\Lambda\to\Lambda,\quad T=T_2\circ T_1,\quad T(\lambda)=\bigcup_{s\in T_1(\lambda)}T_2(s)\tag{13}$$

Before the proof of convergence, several definitions and a theorem[8] are introduced first.

Definition 1 Suppose $\Gamma\subset X$ is a solution set and A is an algorithm on X. A real continuous function Z on X is called an ascent function for Γ and A, if ① $Z(y)>Z(x)$ for $x\notin\Gamma$ and $y\in A(x)$; ② $Z(y)\geqslant Z(x)$ for $x\in\Gamma$ and $y\in A(x)$.

Definition 2 For a point to set mapping $A:X\to Y$, we say A is closed at $x\in X$ if $x_k\to x$, $x_k\in X$ and $y_k\to y$, $y_k\in Y$ imply $y\in A(x)$. We say A is closed on X, if A is closed at all $x\in X$.

Theorem 1 Global convergence theorem

Suppose A is an algorithm on X, let $\{x_k\}_{k=0}^{\infty}$ be the sequence generated by A such that $x_{k+1}\in A(x_k)$, for some $x_0\in X$. $\Gamma\subset X$ is a solution set if ① All the points in $\{x_k\}_{k=0}^{\infty}$ are contained by a compact set $S\subset X$; ② There exists a real continuous function Z on X which is an ascent function for Γ and A; ③ A is closed on $X-\Gamma$. Then the limit of any sub sequence of $\{x_k\}_{k=0}^{\infty}$ is a solution.

To prove the convergence, the following propositions are introduced and proved.

Proposition 1 Suppose $\Gamma\subset\Lambda$ is the fixed points set of mapping T which is defined in Eq. (13), that is, $\Gamma=\{\lambda\mid\lambda\in T(\lambda)\}$. For the given observation sequence $\boldsymbol{O}$, $f:\Lambda\to R$, $f(\lambda)=\max_{(\boldsymbol{Q},\boldsymbol{K})\in S}P(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{O}\mid\lambda)$ is an ascent function for Γ and T.

Proposition 2 The mapping T which is defined in (13) is closed on Λ.

Proposition 3 The algorithm in this paper converges to the fixed points set of the mapping T in Eq. (13).

Proof Theorem 1 gives three conditions for an algorithm's global convergence. For condition ①, we made the assumption of boundedness of Λ which ensures that Λ is compact. For condition ②, we can infer from

proposition 1 that $f: \Lambda \to R, f(\lambda) = \max_{(\boldsymbol{Q},\boldsymbol{K}) \in S} P(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{O} \mid \lambda)$ is an ascent function for Γ and T. As to condition ③, proposition 2 tells us that the mapping T is closed on Λ which ensures its closedness on $\Lambda - \Gamma$. Now, our algorithm in the paper satisfies all three conditions in the theorem, so the algorithm converges to Γ, namely, the fixed points set of the algorithm.

3 Results and Discussion

Our paper tests three algorithms on training of the HMM in face recognition. The generation of observation sequences and selection of features are in accordance with Refs. [9, 10]. That is, scan face images in raster order with a sampling window of size 16 × 16 with 75% overlap to generate observation sequences and take the first 10 significant DCT coefficients of the subimage as features. An ergodic model with three states and each state with two branches is selected in the test and the logarithm method is used to deal with very small probability. The experiment is tested on the ORL database and five face images per person (total 200 images) are used to train the model while the other five face images are used to test. For general purposes, a total of 10 sets are used to train the model, including nine sets selected randomly and one set which has been often used by other references.

The results are shown in Tab. 1, where SB denotes our algorithm. The item of "training set" in the chart refers to the numbers of training images in the ORL database. The implementation of the algorithms has not been optimized, which makes running time a bit longer, but it does not change the relative complexity of three algorithms. As indicated by the results, the three algorithms have close recognition rate but different training time. It takes much more time for BW to train the model than the other two. On average, our algorithm has the best performance on both recognition rate and training time. Furthermore, our algorithm is least sensitive to the training set while the other two depend largely on the training set.

Tab. 1 Results of three algorithms tested on ORL

Number	Training set	Recognition rate/%			Training time/(s·model^{-1})		
		BW	KV	SB	BW	KV	SB
1	1, 2, 3, 4, 5	99.5	98	99	235.9	23.2	48.9
2	1, 8, 7, 10, 6	97.5	96	97.5	199.0	54.1	60.7
3	8, 1, 4, 6, 5	94	94	95	233.2	74.7	54.4
4	2, 4, 8, 10, 1	98.5	98	98.5	278.8	186.1	60.0
5	6, 1, 10, 4, 9	97.5	98	97	330.9	26.1	56.0
6	5, 9, 1, 2, 7	97	97	98	379.1	38.9	49.9
7	1, 7, 8, 3, 5	99	99	99	161.9	102.0	50.6
8	7, 3, 10, 2, 8	97	97	97	167.4	113.1	43.6
9	4, 7, 8, 9, 3	98.5	99	98.5	267.9	83.2	105.1
10	10, 6, 5, 8, 9	95	95	95.5	226.3	12.5	32.5
Average		97.35	97.1	97.5	248.0	71.4	56.2

4 Conclusion

An optimal state and branch sequence based parameter estimation algorithm of continuous HMM with ML as optimal objective is introduced. It is different from the KV algorithm in that only the Viterbi algorithm is used to determine optimal state and branch sequence. When there is only one branch in every state, the two algorithms will predigest to the same one. The results of the three algorithms tested in face recognition indicate that our algorithm achieves the best average performance on recognition rate and training time.

Appendix

Introductory proposition 1 Suppose dual function $f: X \times Y \to R$ is continuous on Y, and $\forall y \in Y$, $f(x, y)$ can attain maximum on X, then function $g: Y \to R$, $g(y) = \max_{x \in X} f(x, y)$ is continuous on Y.

Introductory proposition 2 Suppose dual function $f: X \times Y \to R$ is continuous on both X and Y and M is a point to set mapping where $M(x_0)$, $\forall x_0 \in X$ is the set of all the y which maximize $f(x_0, y)$, then M is closed on X.

Proof of proposition 1 ① For a given observation sequence $\boldsymbol{O}$, $P(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{O} \mid \lambda)$ is a continuous function defined on $S \times \Lambda$. Its continuity on Λ can be seen from formula (10). It is continuous on S because S is a discrete set which has finitely many isolated points. Then we can infer from introductory proposition 1 that $f(\lambda) = \max_{(\boldsymbol{Q},\boldsymbol{K}) \in S} P(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{O} \mid \lambda)$ is continuous on Λ.

② In the iteration of our algorithm we have $\lambda(t) \in T(\lambda(t-1))$. As to $f(\lambda(t)) \geqslant f(\lambda(t-1))$, we have $\max_{(\boldsymbol{Q},\boldsymbol{K})} P(\boldsymbol{O},\boldsymbol{Q},\boldsymbol{K} \mid \lambda(t)) \geqslant^{(1)} P(\boldsymbol{O},\boldsymbol{Q}^*(t-1),\boldsymbol{K}^*(t-1) \mid \lambda(t)) \geqslant^{(2)} P(\boldsymbol{O},\boldsymbol{Q}^*(t-1),\boldsymbol{K}^*(t-1) \mid \lambda(t-1))$ and $P(\boldsymbol{O},\boldsymbol{Q}^*(t-1),\boldsymbol{K}^*(t-1) \mid \lambda(t-1)) = \max_{\boldsymbol{Q},\boldsymbol{K}} P(\boldsymbol{O},\boldsymbol{Q},\boldsymbol{K} \mid \lambda(t-1))$, so $f(\lambda(t)) \geqslant f(\lambda(t-1))$. The inequality (2) is strict unless $\lambda(t-1) \in T(\lambda(t-1))$. That inequality (1) becomes equality implies $(\boldsymbol{Q}^*(t-1), \boldsymbol{K}^*(t$

$-1)) \in T_1(\lambda(t))$ which results in $T_2((\boldsymbol{Q}^*(t-1), \boldsymbol{K}^*(t-1))) \in T_2 \circ T_1(\lambda(t))$, that is, $\lambda(t) \in T(\lambda(t))$. From the two conditions above, it is easy to see that for our algorithm T, the solution set Γ which ensures the ascent of $f(\lambda)$ is the fixed points of T.

Proof of proposition 2 From the proof of proposition 1, for the given observation sequence $\boldsymbol{O}$, $P(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{O} \mid \lambda)$ is a continuous function defined on $S \times \Lambda$. $T_1(\lambda)$ is the set of $(\boldsymbol{Q}, \boldsymbol{K})$ which maximize $P(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{O} \mid \lambda)$, from introductory proposition 2 we known that $T_1: \Lambda \to S$ is closed on Λ. Similarly, $T_2: S \to \Lambda$ is closed on S. Let us prove the closure of T. Suppose that ① $\lambda_1^{(k)} \to \lambda_1$, $\lambda_1 \in \Lambda$, $\lambda_1^{(k)} \in \Lambda$; ② $\lambda_2^{(k)} \to \lambda_2$, $\lambda_2^{(k)} \in T(\lambda_1^{(k)})$. If we can prove $\lambda_2 \in T(\lambda_1)$, then we know that T is closed on Λ. Now, choose $(\boldsymbol{Q}^{(k)}, \boldsymbol{K}^{(k)}) \in T_1(\lambda_1^{(k)})$ so that $\lambda_2^{(k)} \in T_2((\boldsymbol{Q}^{(k)}, \boldsymbol{K}^{(k)}))$, because $(\boldsymbol{Q}^{(k)}, \boldsymbol{K}^{(k)}) \in S$, and for observation sequence of finite length, there are finite points in S. Then $\{(\boldsymbol{Q}^{(k)}, \boldsymbol{K}^{(k)})\}_{k=1}^{\infty}$ must have invariant sub sequence $\{(\boldsymbol{Q}^{(k_i)}, \boldsymbol{K}^{(k_i)})\}_{i=1}^{\infty}$, $(\boldsymbol{Q}^{(k_i)}, \boldsymbol{K}^{(k_i)}) = (Q_0, K_0) \in S$. That T_1 is closed implies $(Q_0, K_0) \in T_1(\lambda_1)$. At the same time, $(\boldsymbol{Q}^{(k_i)}, \boldsymbol{K}^{(k_i)}) \to (Q_0, K_0)$, $\lambda_2^{(k_i)} \in T_2((\boldsymbol{Q}^{(k_i)}, \boldsymbol{K}^{(k_i)}))$, $\lambda_2^{(k_i)} \to \lambda_2$, while T_2 is closed, then $\lambda_2 \in T_2((\boldsymbol{Q}_0, \boldsymbol{K}_0)) \in T_2 \circ T_1(\lambda_1) = T(\lambda_1)$.

References

[1] Rabiner L R. A tutorial on hidden Markov models and selected applications in speech recognition [J]. *Proceedings of the IEEE*, 1989, **77**(2): 257 – 285.

[2] Juang B H, Rabiner L R. The segmental K-means algorithm for estimating parameters of hidden Markov models [J]. *IEEE Trans on Acoustics Speech and Signal Processing*, 1990, **38**(9): 1639 – 1641.

[3] Andrieu C, Doucet A. Simulated annealing for maximum *a posteriori* parameter estimation of hidden Markov models [J]. *IEEE Trans on Information Theory*, 2000, **46**(5): 994 – 1004.

[4] Yishai A B, Burshtein D. A discriminative training algorithm for hidden Markov models [J]. *IEEE Trans on Speech and Audio Processing*, 2004, **12**(5): 204 – 217.

[5] Juang B H, Levinson S E, Sondhi M M. Maximum likelihood estimation for multivariate mixture observations of Markov chains [J]. *IEEE Trans on Information Theory*, 1986, **32**(3): 307 – 309.

[6] Rabiner L R, Juang B H, Levinson S E. Some properties of continuous hidden Markov model representations [J]. *AT& T Technical Journal*, 1985, **64**(6): 1251 – 1269.

[7] Wu C F Jeff. On the convergence properties of the EM algorithm [J]. *The Annals of Statistics*, 1983, **11**(1): 95 – 103.

[8] Chen Baolin. *Optimization theory and algorithm* [M]. Beijing: Tsinghua University Press, 1989. 287 – 299. (in Chinese)

[9] Kohir V V, Desai U B. Face recognition [A]. In: *Proceedings of the IEEE Symposium on Circuits and Systems* [C]. Geneva, Switzerland: Presses Polytechniques et Universitaires Romandes, 2000, **5**: 305 – 308.

[10] Bicego M, Castellani U, Murino V. Using hidden Markov models and wavelets for face recognition [A]. In: Wemer Bob, ed. *Proceedings of the* 12*th International Conference on Image Analysis and Processing* [C]. Mantova, Italy: IEEE Computer Society, 2003. 52 – 56.

基于最优状态和分支序列的连续隐 Markov 模型参数估计

俞 璐[1,2] 吴乐南[1] 谢 钧[3]

([1]东南大学无线电工程系, 南京 210096)

([2]解放军理工大学通信工程学院, 南京 210007)

([3]解放军理工大学指挥自动化学院, 南京 210007)

摘要: 提出了一种连续隐 Markov 模型参数估计算法, 并利用全局收敛定理严格证明了算法的收敛性. 该算法用 Viterbi 算法取代分段 K 平均算法中的聚类方法, 直接确定出最优状态和分支序列, 并依据最优序列以最大似然为优化准则进行参数估计. 阐述了该算法与 Baum-Welch 和分段 K 平均 2 种经典算法在目标函数、优化准则和工作原理等方面的关系, 并将 3 种算法应用于人脸识别. 实验结果表明, 该算法在获得相当识别率的同时缩短了训练时间, 并降低了识别结果对训练样本集的敏感性, 在 3 种算法中总体性能最优.

关键词: 连续隐 Markov 模型; 最优状态和分支序列; 最大似然; 收敛性; Viterbi 算法

中图分类号: TP391