Hidden Markov Models Combining Discrete Symbols and Continuous Attributes in Handwriting Recognition

Hanhong Xue and
Venu Govindaraju, *Senior Member*, *IEEE*

Abstract—Prior arts in handwritten word recognition model either discrete features or continuous features, but not both. This paper combines discrete symbols and continuous attributes into structural handwriting features and model, them by transition-emitting and state-emitting hidden Markov models. The models are rigorously defined and experiments have proven their effectiveness.

Index Terms—Markov processes, handwriting analysis.

1 Introduction

Stochastic models, especially hidden Markov models (HMMs), have been successfully applied to the field of offline handwriting recognition in recent years. These models can generally be categorized as being either discrete or continuous, depending on their observation types.

Bunke et al. [1] model an edge in the skeleton of a word image by its spatial location, degree, curvature, and other details, and derive 28 symbols by vector quantization for discrete HMMs Chen et al. [2] use 35 continuous features including momental, geometrical, topological, and zonal features in building continuous density and variable duration HMMs. Mohammed and Gader [3] incorporate locations of vertical background-foreground transitions in their continuous density HMMs. Senior and Robinson [4] describe a discrete HMM system modeling features extracted from a grid. The features include information such as the quantized angle that a stroke makes when entering from one cell to another and the presence of dots, junctions, endpoints, turning points, and loops in a cell. El-Yacoubi et al. [5] adopt two sets of discrete features: global features (loops, ascenders, descenders, etc.) and bidimensional dominant transition numbers, in their HMMs.

As can be seen, most of the previous studies dealing with stochastic models focus on modeling low-level statistical features and are either discrete or continuous. In studying handwriting recognition using high-level structural features, such as loops, crosses, cusps, and arcs shown in Fig. 1a, we find it more useful to associate these features, which are discrete symbols, with some continuous attributes including position, orientation, and angle between strokes as shown in Fig. 1b. For example, vertical position is critical in distinguishing an "e" and an "l" when they are both written in loops. This paper explores a new approach to modeling sequences consisting of discrete symbols and their continuous attributes for offline handwriting recognition. Our approach has been applied to the construction of word recognizers based on variants of HMMs, showing positive effect of using continuous attributes and promising results in handwritten word recognition.

An HMM can emit output from either transitions (Mealy machine) or states (Moore machine). Bourlard and Bengio [6] summarize the differences between a variety of stochastic models that are based on Markov model, HMM, HMM/ANN (Aritificial Neural Network) hybrid, and Input/Output HMM as well as the difference between their transition-emitting (Mealy) and state-emitting (Moore) variants. We will tailor both transition-emitting HMMs (TE-HMMs) and state-emitting HMMs (SE-HMMs) for the purpose of modeling discrete symbols and their continuous attributes together.

This paper is organized as follows: Section 2 describes the structural features to model. Section 3 defines TE-HMMs to model the sequences of structural features, giving its training and decoding algorithms. In Section 4, we define SE-HMMs and derive algorithms for SE-HMMs directly from those for TE-HMMs. We build two new word recognizers based on TE-HMMs and SE-HMMs in Section 5 and present experimental results on the effect of using continuous attributes in addition to discrete symbols, on the comparison of the new recognizers against other recognizers tested on the same image set. Finally, Section 6 concludes this paper and proposes future directions.

2 Structural Features

It appears from psychological studies [7], [8] that word shape plays a significant role in visual word recognition, which inspires us to investigate the use of shape-defining features, i.e., high-level structural features, in building word recognizers. As widely used in holistic paradigms, ascenders and descenders are prominently shape-defining. However, there are many cases where words do not have ascenders and descenders, demanding other structural features.

An oscillation handwriting model was investigated by Hollerbach [9]. During writing, the pen moves from left to right horizontally and oscillates vertically. The study has shown that features near vertical extrema are very important in character shape definition. These features include ascenders, descenders, loops, crosses, turns, and ends. Since the concepts of ascender and descender actually indicate the position of structures rather than the structures themselves, position becomes an important attribute of structural features. Besides position, a structure feature can have other attributes such as orientation, curvature, and size, as illustrated in Fig. 1b. Once we are able to utilize structural features together with their attributes in defining the shape of characters and, thus, the shape of words, we can construct a recognizer that simulates the human's shape discerning capability in visual word recognition.

Dzuba et al. [10] have noted the importance of structural features and built in a high-performance word recognizer. However, only discretized vertical position is considered besides the discrete name of a structural feature, and the training of a character model is done manually based on intuition and by trail-and-error. To make the training of word recognizers automated, this paper applies TE-HMMs and SE-HMMs to accommodate structural features, considering both their discrete names and their continuous attributes.

Table 1 lists the structural features that are used to model handwriting in this paper. Among them, loops, arcs, and gaps are straightforward as illustrated by Fig. 1. Long cusps and short cusps are separated by thresholding their vertical length. Left/right-ended arcs are arcs whose stroke ends at its left/right side. Circles are typically in letter "o" and crosses in letter "t." Bars are likely to be in uppercase letters, "T," for example. Loops, arcs, and cusps are further divided into upward ones and downward ones as shown in Fig. 1a.

For each feature, there is a set of continuous attributes associated with it. For example, position of a feature is an attribute that is measured relative to reference lines. Orientation and angle are measured in degrees. The width of a feature is measured relative to the average width of a character. All the features and their attributes are extracted by the skeletal graph approach described in [11], where heuristics are applied to order the

- H. Xue is with the Advance Clustering Technology Team, IBM, 2455 South Road, Poughkeepsie, NY 12601. E-mail: hanhong@us.ibm.com.
- V. Govindaraju is with the Center for Document Analysis and Recognition, UB Commons, Suite 202, Amherst, NY 14228. E-mail: venu@cubs.buffalo.edu.

Manuscript received 10 July 2003; revised 13 May 2005; accepted 20 July 2005; published online 13 Jan. 2006.
Recommended for acceptance by L. Vincent.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0168-0703.

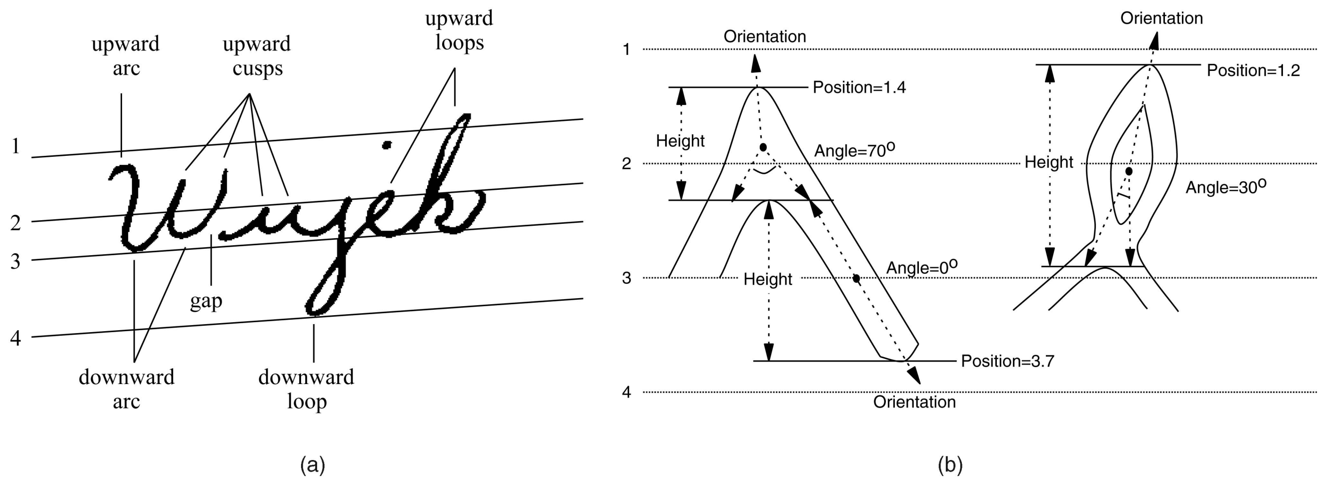

Fig. 1. High-level structural features and their continuous attributes. (a) Features. (b) Attributes.

structural features roughly in the same order as they are naturally written and no segmentation is required.

Suppose a structural feature is given as (u, v), where u is the feature type and v is a vector of attributes associated with the type. The probability of having (u, v) can be decomposed into two parts $P(u, v) = P(u)P(v|u)$, where the distribution of $P(u)$ is discrete and that of $P(v|u)$ is continuous. Therefore, $P(u)$ can be modeled by discrete probabilities and $P(v|u)$ can be modeled by multivariate Gaussian distributions. The advantage of such a decomposition is that each feature type can have a different number of attributes.

3 Transition-Emitting HMMs

3.1 Definition

To model sequences of structural features which are symbols associated with continuous attributes, we define a transition-emitting HMM (TE-HMM) as $\lambda = < S, L, A >$. $S = \{s_1, s_2, \dots, s_N\}$ is a set of states, assuming a single starting state s_1 and a single accepting state s_N. $L = \{l_1, l_2, \dots, l_M\}$ is a set of discrete symbols corresponding to feature types. For each symbol, there is a set of continuous attributes that describes its details. So, an observation is represented as $o = (u, v)$, where $u \in L$ is a symbol and v a vector of continuous values. A special null symbol ϵ has no attributes and does not appear in the input. $A = \{a_{ij}(o)\}$, the *observation probability*, is a set of probability density functions (pdfs), where $a_{ij}(o)$ is the pdf of features observed while transitioning from state i to state j. The sum of outgoing probabilities from a state must be 1, i.e., $\sum_j [a_{ij}(\epsilon) + \sum_u \int_v a_{ij}(u, v)dv] = 1$ for all state i.

The special null symbol ϵ is used to account for missing features that are dropped due to sloppy writing or imperfect feature extraction It allows a model to transition from one state to another without observing/consuming any input feature, so that the model is able to accept a feature sequence that is shorter than expected. Also, we allow a model to have self-transitions from a state into the state itself observing nonnull symbols. In this way, the model is able to accept a feature sequence that is longer than expected. The null symbol and nonnull symbols will be treated differently because of the fact that the null symbol never presents in the input.

By the above definition, the probability of transitioning from state i to state j while observing an input feature o is the observation probability $a_{ij}(o)$. For a nonnull observation $o = (u, v) = (l_k, v)$, the observation probability $a_{ij}(o)$ is decomposed into two parts: $a_{ij}(o) = P(l_k, v|i, j) = P(l_k|i, j)P(v|l_k, i, j) = f_{ij}(l_k)g_{ijk}(v)$. The first part $f_{ij}(l_k) = P(l_k|i, j)$ will be referenced as the *symbol observation probability*, which is the probability of observing a symbol l_k regardless of its attributes on the transition from state i to state j. The second part $g_{ijk}(v) = P(v|l_k, i, j)$ will be reference as the *attribute observation probability*, which is the distribution of symbol l_k's attributes on the transition form state i to state j. The special null symbol does not have any attribute, so its observation probability is simply denoted as $a_{ij}(\epsilon) = P(\epsilon|i, j) = f_{ij}(\epsilon)$, where only the symbol observation probability is present. Then, by the nature of probability, we have $f_{ij}(\epsilon) + \sum_k f_{ij}(l_k) = 1$ and $\int_v g_{ijk}(v)dv = 1$ which will be the guidelines of reestimation.

We model g_{ijk} by multivariate Gaussian distributions $\mathcal{N}(\mu_{ijk}, \sigma_{ijk})$, where μ_{ijk} is the average of attributes of symbol l_k on the transition from state i to state j and σ_{ijk} is the covariance matrix of these attributes In practice, we assume the covariance matrix is a diagonal matrix for simplicity and that attributes involved are strongly independent of each other. It should be noted that symbols are not required to have the same number of attributes. As the number of attributes increases, observation probabilities decrease exponentially. Therefore, they are actually normalized by taking their $d_k\text{th}$ root to make them comparable where d_k is the number of attributes symbol l_k has. This normalization takes the geometric mean of the probabilities of attributes and is done consistently in both training and decoding. Without this normalization, the model will be biased to the features with less number of attributes.

3.2 Observation Sequence and State Sequence

Observation sequences of structural features are extracted from handwriting images for training and testing. An observation sequence is defined as $O = (o_1, o_2, \dots, o_T)$, where $o_t = (u_t, v_t)$, $u_t \in L$, and v_t is a vector of continuous values. The transition from state i at time t to state j at time $t+1$ is decided by the input observation o_t and the probability is the observation probability $a_{ij}(o_t)$. Also, the transtion from state i at time t to state j at time t is possible when the null symbol is allowed on the transition and the probability is $a_{ij}(\epsilon)$.

We use $Q(t, i)$ to denote that the model is in state i at time t. Given an input observation sequence $O = (o_1, o_2, \dots, o_T)$, a state sequence $Q(t_0, q_0), Q(t_1, q_1), \dots, Q(t_W, q_W)$ describes how the model interprets

TABLE 1
Structural Features and Their Attributes

structural features	pos.	orient.	angle	width
short cusp, long cusp	X	X		
arc, left-ended arc, right-ended arc	X		X	
loop, circle, cross, bar	X			
gap				X

the input by transitioning from the starting state at time 0 to the accepting state at time T. So, it is required that $t_0 = 0$, $q_0 = 1$, $t_W = T$, and $q_W = N$. Assuming first-order Markov property in the state sequence, i.e., the next state depends on the current state but not past states, we define the probability of a state sequence as the product of observation probabilities on its transitions. The probability of an input observation sequence given a model $P(O|\lambda)$ is the sum of the probabilities of all possible state sequences to interpret O on λ. Forward and backward probabilities are applied to the calculation of $P(O|\lambda)$, which is discussed in the next section.

3.3 Forward and Backward Probabilities

The training is done by the Forward-Backward algorithm [12], with a little modification. The forward probability $\alpha_j(t) = P(o_1, o_2, \dots o_t, Q(t, j)|\lambda)$ is defined as the probability of being in state j after having observed the first t observations given the model and the backward probability $\beta_i(t) = P(o_{t+1}, o_{t+2}, \dots o_T, Q(t, i)|\lambda)$ as the probability of being in state i before the last $T - t$ observations given the model. $\alpha_N(T) = \beta_1(0) = P(O|\lambda)$ is the overall probability of having the input given the model. $\alpha_j(t)$ and $\beta_i(t)$ are calculated as

$$\alpha_j(t) = \begin{cases} 1 & , j = 1, t = 0 \\ \sum_i (\alpha_i(t) a_{ij}(\epsilon) + \alpha_i(t-1) a_{ij}(o_t)) & , \text{otherwise} \end{cases} \quad (1)$$

$$\beta_i(t) = \begin{cases} 1 & , i = N, t = T \\ \sum_j (a_{ij}(\epsilon) \beta_j(t) + a_{ij}(o_t) \beta_j(t+1)) & , \text{otherwise.} \end{cases} \quad (2)$$

The first term in the sum is for observing the null symbol, which does not consume any input observation and the second term is for observing some nonnull symbol in the input.

3.4 Reestimation

The reestimation process is a variant of the Forward-Backward algorithm. Suppose the model to be trained is λ and the single training example is $O = (o_1, o_2, \dots, o_T)$. We feed the example to the model and calculate the likelihood $P(O|\lambda)$ using the forward and backward probabilities. Obviously, different transitions contribute differently to $P(O|\lambda)$ during the calculation. If observation probabilities are adjusted according to their contributions to $P(O|\lambda)$, then the model is adapted to the example. In this learning process, the probability of observing o while transitioning from state i to state j can be reestimated as the number of times observing o while transitioning from state i to state j divided by the total number of times transitioning from state i, according to the constraint that a state's outgoing probabilities must sum to 1. Once we have reestimated observation probabilities, the training example O can be fed to the model λ again for yet another reestimation and this can be done iteratively until the reestimation reaches a local maximum of $P(O|\lambda)$.

As before, the null symbol and nonnull features are treated differently. We define two probabilities, $\omega_{ij}(t) = P(Q(t, i), Q(t, j)|O, \lambda)$ as the probability of observing ϵ while transitioning from state i to state j at time t and $\tau_{ij}(t) = P(Q(t-1, i), Q(t, j)|O, \lambda)$ as the probability of observing a nonnull symbol o_t while transitioning from state i at time $t - 1$ to state j at time t. $\omega_{ij}(t)$ and $\tau_{ij}(t)$ can be computed by

$$\begin{aligned} \omega_{ij}(t) &= P(Q(t, i), Q(t, j)|O, \lambda) \\ &= \frac{P(Q(t, i), Q(t, j), O|\lambda)}{P(O|\lambda)} = \frac{\alpha_i(t) a_{ij}(\epsilon) \beta_j(t)}{\alpha_N(T)} \\ \tau_{ij}(t) &= P(Q(t-1, i), Q(t, j)|O, \lambda) \\ &= \frac{P(Q(t-1, i), Q(t, j), O|\lambda)}{P(O|\lambda)} = \frac{\alpha_i(t-1) a_{ij}(o_t) \beta_j(t)}{\alpha_N(T)}. \end{aligned} \quad (3)$$

The symbol observation probability $f_{ij}(u)$ is reestimated as the expected number of transitions from state i to state j seeing symbol u divided by the expected number of transitions out from state i, as given by (4). This estimation directly conforms to the constraint that the sum of symbol observation probabilities on a transition must be 1.

$$\hat{f}_{ij}(u) = \begin{cases} \frac{\sum_t \omega_{ij}(t)}{\sum_j \sum_t (\omega_{ij}(t) + \tau_{ij}(t))} & , u = \epsilon \\ \frac{\sum_{t, u_t = u} \tau_{ij}(t)}{\sum_j \sum_t (\omega_{ij}(t) + \tau_{ij}(t))} & , u \neq \epsilon. \end{cases} \quad (4)$$

Since the null symbol does not have any attribute, reestimation of attribute observation probability is only necessary for nonnull symbols. The average of attributes of symbol l_k on the transition from state i to state j is reestimated as

$$\hat{\mu}_{ijk} = \frac{\sum_{t, u_t = l_k} \tau_{ij}(t) v_t}{\sum_{t, u_t = l_k} \tau_{ij}(t)}, \quad (5)$$

which is the weighted average of attribute values across the time when l_k is observed. Similarly, the covariance of these attributes is reestimated as

$$\hat{\sigma}_{ijk} = \frac{\sum_{t, u_t = l_k} \tau_{ij}(t) (v_t - \mu_{ijk})' (v_t - \mu_{ijk})}{\sum_{t, u_t = l_k} \tau_{ij}(t)}, \quad (6)$$

which is the weighted average of attribute covariances across the time when l_k is observed.

3.5 Parameter Tying

Parameter tying has been widely applied in speech recognition [13]. In our case, self-transitions in a model have the ability of absorbing extra features and they are more likely to observe all different kinds of features with large variations in attributes and the estimation of their parameters tends to be less reliable. Therefore, the attributes of a feature are tied for all self-transitions in a model in reestimation and shared in decoding. Let μ_k and σ_k be the mean and the variance of the attributes of l_k on all self-transitions, respectively. They are reestimated by

$$\hat{\mu}_k = \frac{\sum_i \sum_{t, u_t = l_k} \tau_{ii}(t) v_t}{\sum_i \sum_{t, u_t = l_k} \tau_{ii}(t)} \quad (7)$$

and

$$\hat{\sigma}_k = \frac{\sum_i \sum_{t, u_t = l_k} \tau_{ii}(t) (v_t - \mu_k)' (v_t - \mu_k)}{\sum_i \sum_{t, u_t = l_k} \tau_{ii}(t)}, \quad (8)$$

where τ_{ii} indicates self-transition and $\sum_i$ means that all self-transitions are tied.

3.6 Decoding

The decoding is done by the Viterbi algorithm [14], which produces the most probable state sequence for a given input O. Define $\gamma_j(t)$, the Viterbi probability, as the highest probability of being in state j at time t produced by one state sequence.

$$\gamma_j(t) = \begin{cases} 1 & , j = 1, t = 0 \\ \max(\max_i \gamma_i(t) a_{ij}(\epsilon), \max_i \gamma_i(t-1) a_{ij}(o_t)) & , \text{otherwise,} \end{cases} \quad (9)$$

where $\max_i \gamma_i(t) a_{ij}(\epsilon)$ is the highest probability resulting from observing a null symbol when entering state j from state i and $\max_i \gamma_i(t-1) a_{ij}(o_t)$ is the highest probability resulting from observing the nonnull symbol o_t when entering the state j from state i. Finally, $\gamma_N(T)$ is the Viterbi probability of observing the entire sequence O given the model.

4 State-Emitting HMMs

4.1 Definition

Observations are emitted on transitions in TE-HMMs and they can also be emitted from states. A state-emitting HMM (SE-HMM) is defined as $\lambda = < S, L, B, C >$. S and L follow the same definitions as in TE-HMM. $B = \{b_{ij}\}$ is a set of *transition probabilities* where b_{ij} is the probability of transitioning from state i to state j. The sum of transition probabilities from a state must be 1, i.e., $\sum_j b_{ij} = 1$ for all i. $C = \{c_j(o)\}$ is a set of *emission probabilities* where $c_j(o)$ is the probability of observing $o = (u, v)$ on state j. The sum of emission probabilities on a state must be 1, i.e., $c_j(\epsilon) + \sum_u \int_v c_j(u, v)dv = 1$ for all state j.

The emission probability of a non-null symbol $o = (u, v) = (l_k, v)$ is decomposed into two parts: $c_j(o) = P(l_k, v|j) = P(l_k|j)P(v|l_k, j) = f_j(l_k)g_{jk}(v)$, where $f_j(l_k) = P(l_k|j)$ is the *symbol emission probability* and $g_{jk}(v) = P(v|l_k, j)$ is the *attribute emission probability*. For the null symbol, since it does not have any attribute, its symbol emission probability is denoted by $c_j(\epsilon) = P(\epsilon|j) = f_j(\epsilon)$ and there is no attribute emission probability for it. By the nature of probability, we have $f_j(\epsilon) + \sum_k f_j(l_k) = 1$ and $\int_v g_{jk}(v)dv = 1$. We model g_{jk} by multivariate Gaussian distributions $\mathcal{N}(\mu_{jk}, \sigma_{jk})$, where μ_{jk} is the average of attributes of symbol l_k on state j and σ_{jk} is the covariance matrix of these attributes.

A SE-HMM $< S, L, B, C >$, where $B = \{b_{ij}\}$ and $C = \{c_j(o)\}$ can be easily converted into a TE-HMM $< S, L, A >$ by defining $A = \{a_{ij}(o)\} = \{b_{ij}c_j(o)\}$. The constraint that all out-going observation probabilities of a state must be 1 still holds as

$$\sum_j \Big[a_{ij}(\epsilon) + \sum_u \int_v a_{ij}(u, v)dv \Big] = \sum_j \Big[b_{ij}c_j(\epsilon) + b_{ij} \sum_u \int_v c(u, v)dv \Big] = \sum_j b_{ij} = 1.$$

Hence, from the equality $a_{ij}(o) = b_{ij}c_j(o)$, we can derive forward and backward probabilities from (1) and (2), $\omega_{ij}(t)$ and $\tau_{ij}(t)$ from (3), and Viterbi decoding from (9). Only reestimation of B and C are slightly different, as explained below.

4.2 Reestimation

The transition probability b_{ij} is reestimated as the expected number of transitions from state i to state j divided by the expected number of transitions out from state i, i.e.,

$$\hat{b}_{ij} = \frac{\sum_t (\omega_{ij}(t) + \tau_{ij}(t))}{\sum_j \sum_t (\omega_{ij}(t) + \tau_{ij}(t))}. \quad (10)$$

This equation conforms to the constraint that the sum of all outgoing transition probabilities from a state must be 1.

The symbol emission probability $f_j(u)$ is reestimated as

$$\hat{f}_j(u) = \begin{cases} \frac{\sum_i \sum_t \omega_{ij}(t)}{\sum_i \sum_t (\omega_{ij}(t) + \tau_{ij}(t))} & , u = \epsilon \\ \frac{\sum_i \sum_{t, u_t = u} \tau_{ij}(t)}{\sum_i \sum_t (\omega_{ij}(t) + \tau_{ij}(t))} & , u \neq \epsilon. \end{cases} \quad (11)$$

Here, we need to consider all incoming transitions to state j and all symbol emission probabilities on a state must sum up to 1.

Since the null symbol does not have any attribute, reestimation of attribute emission probability is only necessary for nonnull symbols. The average of attributes of symbol l_k on state j is reestimated as

$$\hat{\mu}_{jk} = \frac{\sum_i \sum_{t, u_t = l_k} \tau_{ij}(t) v_t}{\sum_i \sum_{t, u_t = l_k} \tau_{ij}(t)} \quad (12)$$

and the covariance of these attributes is similarly reestimated as

$$\hat{\sigma}_{jk} = \frac{\sum_i \sum_{t, u_t = l_k} \tau_{ij}(t)(v_t - \mu_{jk})'(v_t - \mu_{jk})}{\sum_i \sum_{t, u_t = l_k} \tau_{ij}(t)}. \quad (13)$$

SE-HMMs do not have any observations on transitions, so parameter tying for attributes on self-transitions, as we did for TE-HMMs, is not applicable here.

5 Experimental Results

We implement the above described stochastic models for handwritten word recognition. Character models, including both uppercase and lowercase, are built from training feature sequences extracted on character images. The number of states in a model is simply decided by the average length of the training sequences and the transitions are only allowed from state i to state j if 1) $j = i$ or 2) $j > i$ and $j - i \equiv 1 \bmod 2$. Both TE-HMM and SE-HMM use the same topology for the same character. Word models are obtained by concatenating character models with bigram case probabilities, which are the probabilities of having the case of a character given the case of its previous character A trailing transition is appended to character models to model the gaps between characters. All models are initialized with flat probabilities for symbols and arbitrary Gaussian distributions for attributes, e.g., $\mathcal{N}(2, 1)$ for position and $\mathcal{N}(120^o, 60^o)$ for angle.

We use 5,596 US city names and state names in CEDAR (Center of Excellence for Document Analysis and Recognition) CDROM [15] for training and the 3,000 US city names and state names in CEDAR BHA data set for testing. The BHA testing set is considered as relatively difficult because some words that are very similar to the truth have been inserted in the lexicon to deliberately confuse the recognizers.

5.1 Effect of Continuous Attributes

In order to test the effectiveness of associating continuous attributes with discrete symbols, we start without any continuous attributes and add them one by one. The first attribute added is the width of gaps and the position of all other structures. The second attribute added is the orientation of cusps and the angle of arcs. It should be noted that some features, such as gaps and loops, do not have more than one attribute, so effectively we are modeling features with different numbers of attributes. Table 2 shows accuracy rates with lexicons of different sizes. It can be seen that the addition of continuous attributes significantly improves the recognizer's performance, especially when the lexicon size is large.

5.2 Comparison with Other Recognizers

Table 3 compares the stochastic recognizers against other recognizers described in the literature. All the recognizers are trained with CEDAR CDROM datasets and tested with CEDAR BHA data sets. The first one is a recognizer modeling image segments by continuous density variable duration HMMs [16]. The second one is an approach of oversegmentation followed by dynamic programming on segment combinations [17]. The third one is a recently improved version of the second one by incorporating Gaussian mixtures to model character clusters [18]. The stochastic recognizers described compare very favorably against [16] and [17], and are only slightly worse than [18].

For small lexicons, there is no significant difference between TE-HMMs and SE-HMMs. However, for larger lexicons, the advantage of TE-HMMs becomes evident due to more parameters in TE-HMMs, which improve the modeling power. On the other hand, since SE-HMMs based on the same model topology have fewer parameters to train, they could be more advantageous when the amount of training data is not sufficient to train TE-HMMs.

TABLE 2
Recognition Results with 0, 1, and 2 Continuous Attributes

Lexicon	In top	SE-HMM attributes			TE-HMM attributes		
size	choices	0	1	2	0	1	2
10	1	93.59	94.96	96.36	94.46	95.66	96.56
	2	97.43	97.83	98.60	97.96	98.19	98.77
100	1	75.67	82.08	86.35	80.14	85.15	89.12
	2	85.35	89.79	92.66	88.28	91.56	94.06
	10	96.90	97.36	98.36	96.90	97.93	98.19
	20	98.63	98.97	99.17	98.77	98.83	99.10
1000	1	56.16	64.97	70.97	62.56	69.97	75.38
	2	67.97	77.78	82.78	74.87	82.68	86.29
	10	84.18	91.10	93.59	88.19	92.29	94.39
	20	90.09	93.99	96.17	91.99	94.59	96.50
	100	98.00	98.39	99.20	97.70	99.60	99.10
20000	1	32.18	44.39	51.13	38.35	50.40	58.14
	2	41.26	54.54	60.15	48.10	59.15	66.49
	10	60.38	71.13	77.30	66.76	75.63	81.31
	20	67.52	77.10	82.74	73.60	80.41	85.71
	100	82.31	88.35	91.59	86.45	89.79	93.39

TABLE 3
Performance Comparison to Other Three Word Recognizers on the Same Testing Set

Size		[16]	[17]	[18]	SE-HMM	TE-HMM
10	Top 1	93.2	96.80	96.86	96.36	96.56
	Top 2		98.63	98.80	98.60	98.77
100	Top 1	80.6	88.23	91.36	86.35	89.12
	Top 2		93.36	95.30	92.66	94.06
	Top 3	90.2				
	Top 10			98.53	98.36	98.19
	Top 20		98.93	99.07	99.17	99.10
1000	Top 1	63.0	73.80	79.58	70.97	75.38
	Top 2		83.20	88.29	82.78	86.29
	Top 3	79.3				
	Top 5	83.9		93.29	90.29	91.69
	Top 10			95.50	93.59	94.39
	Top 50		98.70	98.00	98.30	98.40
20000	Top 1			62.43	51.13	58.14
	Top 2			71.07	60.15	66.49
	Top 10			83.62	77.30	81.31
	Top 20			87.49	82.74	85.71
	Top 100			93.59	91.59	93.39

6 Conclusions and Future Work

This paper presents a stochastic framework for modeling features that consist of discrete symbols associated with continuous attributes, aimed at applications to offline handwritten word recognition using high-level structural features. In this framework, different sets of attributes can be associated with different discrete symbols, providing variety and flexibility in modeling details. As supported by experiments, the addition of continuous attributes to discrete symbols does improve the overall recognition accuracy significantly Now, we are investigating some possible ways of improving the recognizer. The first one is to expand the feature set to capture more detailed handwriting styles, which means more symbols and more attributes. The second one is to make feature extraction more robust to writing styles and image quality. The third one is to optimize the model topology by learning from training examples. All these remain challenging tasks for the future.

References

[1] H. Bunke, M. Roth, and E. Schukat-Talamazzini, “Off-Line Cursive Handwriting Recognition Using Hidden Markov Models,” *Pattern Recognition,* vol. 28, no. 9, pp. 1399-1413, 1995.

[2] M. Chen, A. Kundu, and S. Srihari, “Variable Duration Hidden Markov Model and Morphological Segmentation for Handwritten Word Recognition,” *IEEE Trans. Image Processing,* vol. 4, no. 12, pp. 1675-1688, 1995.

[3] M. Mohammed and P. Gader, “Handwritten Word Recognition Using Segmentation-Free Hidden Markov Modeling and Segmentation-Based Dynamic Programming Techniques,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 5, pp. 548-554, May 1996.

[4] A. Senior and A. Robinson, “An Off-Line Cursive Handwriting Recognition System,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 3, pp. 309-321, Mar. 1998.

[5] A. El-Yacoubi, M. Gilloux, R. Sabourin, and C.Y. Suen, “An HMM-Based Approach for Off-Line Unconstrained Handwritten Word Modeling and Recognition,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 8, pp. 752-760, Aug. 1999.

[6] H. Bourlard and S. Bengio, “Hidden Markov Models and Other Finite State Automata for Sequence Processing,” *The Handbook of Brain Theory and Neural Networks,* second ed. Cambridge, Mass.: MIT Press, 2002.

[7] D. Wheeler, “Word Recognition Processes,” *Cognitive Psychology,* vol. 1, pp. 59-85, 1970.

[8] G. Humphreys, “Orthographic Processing in Visual Word Recognition,” *Cognitive Psychology,* vol. 22, pp. 517-560, 1990.

[9] J. Hollerbach, “An Oscillation Theory of Handwriting,” *Biological Cybernetics,* vol. 39, pp. 139-156, 1981.

[10] G. Dzuba, A. Filatov, D. Gershuny, and I. Kil, “Handwritten Word Recognition—The Approach Proved by Practice,” *Proc. Sixth Int'l Workshop Frontiers in Handwriting Recognition,* pp. 99-111, 1998.

[11] H. Xue and V. Govindaraju, “Building Skeletal Graphs for Structural Feature Extraction on Handwriting Images,” *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 96-100, 2001.

[12] L. Baum, “An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes,” *Inequalities,* vol. 3, pp. 1-8, 1972.

[13] S. Young, “Large Vocabulary Continuous Speech Recognition: A Review,” *Proc. IEEE Workshop Automatic Speech Recognition and Understanding,* pp. 3-28, 1995.

[14] G. Forney, “The Viterbi Algorithm,” *Proc. IEEE,* vol. 61, no. 3, pp. 263-278, 1973.

[15] J.J. Hull, “A Database for Handwritten Text Recognition Research,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 5, pp. 550-554, May 1994.

[16] M. Chen, “Handwritten Word Recognition Using Hidden Markov Models,” PhD thesis, State Univ. of New York at Buffalo, 1993.

[17] G. Kim and V. Govindaraju, “A Lexicon Driven Approach to Handwritten Word Recognition for Real-Time Applications,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 4, pp. 366-379, Apr. 1997.

[18] S. Tulyakov and V. Govindaraju, “Probabilistic Model for Segmentation Based Word Recognition with Lexicon,” *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 164-167, 2001.