2011 International Conference on Document Analysis and Recognition

Concurrent Optimization of Context Clustering and GMM for Offline Handwritten Word Recognition Using HMM

Tomoyuki Hamamura[1,2], Bunpei Irie[1], Takuya Nishimoto[2], Nobutaka Ono[2] and Shigeki Sagayama[2]
[1]*TOSHIBA Corporation, Tokyo, Japan*
[2]*Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan*
[1] *tomoyuki.hamamura@toshiba.co.jp*

***Abstract*—Context-dependent HMMs are commonly used in speech recognition. Parameter sharing needed for this model can be realized by two methods: context clustering or tied-mixture. In speech recognition, the former is reported to be more precise. However, there is some difficulty in applying context clustering to handwritten word recognition, since the distribution of each character is typically a mixture of different distributions, such as block-printed, cursive, etc. For this reason, successful results reported so far are limited to the tied-mixture approach. To deal with this problem, we propose a novel parameter tying method "Partial Tied-Mixture", where the Gaussian Mixture Model (GMM) consists of a portion of all Gaussians. Furthermore, we derive a method to concurrently optimize context clustering and GMM. Experiments on the CEDAR database show that the proposed method outperforms tied-mixture both in terms of precision and computational cost.**

***Keywords*-Handwritten word recognition; Context clustering; GMM; Context-dependent HMM; Partial Tied-Mixture; EM algorithm;**

I. Introduction

Context-dependent HMMs are a commonly used technique in speech recognition. Tuning the parameters for each context improves the recognition rate, since a speech waveform is distorted, depending on the preceding and succeeding phoneme, or the context. The same kind of distortion occurs in handwriting. However, to the best of our knowledge, there have been only two attempts at online handwriting recognition based on context-dependent HMMs [1], [2], and five at offline [3], [4], [5], [6], [7].

A context-dependent model has a risk of overfitting, due to an increase in the number of parameters. Tying parameter values among models is necessary to prevent this. There are two methods for parameter tying: context clustering and tied-mixture. Context clustering approach subdivides the context into clusters and shares the parameters among the contexts in the same cluster. Contexts are typically clustered by the state position of each phoneme model, and the states are shared among the contexts in the same cluster. This method usually provides higher precision compared with the tied-mixtures.

On the other hand, tied-mixture approach shares the Gaussian of the GMM among states, only varying the mixing coefficients. Natarajan et al. reports that sharing Gaussians only among the same character and the same state position leads to a higher recognition rate [3]. This method is usually employed in order to improve the efficiency, since the degradation of the precision is relatively small for a small number of mixture components.

There are two types of context clustering: those which utilize a prior knowledge on the recognition target, and those which do not. Tree-Based Clustering, often employed in speech recognition, is of the former type. This type needs a specialized knowledge on the recognition target, which prevents it from being applied to various targets, such as different languages (ex. Arabic, Korean), different sequential data type (ex. handwriting, speech, sign language), and so on. Furthermore, sometimes it cannot be used when feature or model configuration are changed. Therefore, a context clustering method which does not need specialized knowledge has been eagerly awaited.

Since context clustering provides a high precision as noted above, there is an expectation that this technique can be employed in offline handwriting recognition. However, successful context clustering without using prior knowledge on the recognition target has not been reported to date. To the best of our knowledge, there have been only 2 works dealing with context-dependent models without prior knowledge [3], [4]. Both of them adopts tied-mixture approach. There have been 3 published works discussing context clustering [5], [6], [7], but all of them use prior knowledge on the recognition target. Fink et al. reports that context clustering without using prior knowledge lowers the recognition rate. The reason for this is thought to be the fact that the variation of the letter form of a single category caused by the writing style, like block print or script, is far greater than the fluctuation due to the context.

In this paper, a novel method for parameter tying – Partial Tied-Mixture (PTM) – is proposed. PTM is the generalization of the conventional context clustering, and it is effective even in the cases with greater letter shape variation within a single category. We show that the optimization of PTM's parameters can be seen as a concurrent optimization of context clustering and GMM. The problems of conventional methods are pointed out in section II. PTM is presented, and the parameter updating formula is derived out in section III. The effectiveness of PTM is demonstrated in section IV.

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.111

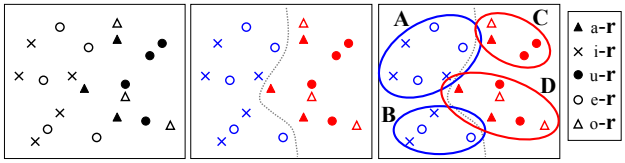

Figure 1. An example of the distribution of phoneme r for each context of preceding phonemes

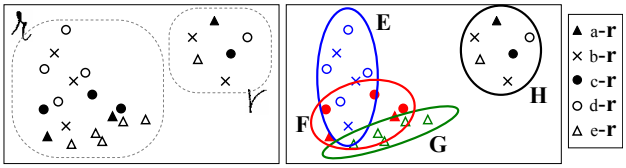

Figure 2. An example of the distribution of character r for each context of preceding characters

II. Problems of Conventional Methods

The conventional model learning flow using context clustering is as follows:

Step 1: Divide the context by using some clustering technique for each state position of each phoneme.

Step 2: Let all the context in the same cluster share the states.

Step 3: Estimate GMM parameters for the shared state.

Fig.1(a) is an example of the distribution of the feature vector corresponding to a particular state position of a particular phoneme ("r") for each preceding phoneme (a-, i-, u-, e-, o-). Correspondence between state and feature vector can be found by using Baum-Welch algorithm or Viterbi algorithm. Fig.1(b) is the result of step1, showing two clusters (i-r,e-r) and (a-r,u-r,o-r). Fig.1(c) is the result of step3, in which each cluster is estimated by 2 Gaussians. The conventional learning step works well for such cases with comparatively large variation by context.

In the case of handwritten character recognition, however, each character has distinctively different letter shapes like block print or cursive scripts, which leads to larger variation than that which is caused by the context. Fig.2(a) is an example of the distribution of handwritten "r" for each context of the preceding character (a-, b-, c-, d-, e-). The distribution is distinctly divided into two parts as illustrated, since the shapes of block print and cursive script differ largely from each other. In these cases, the effect of the context results in a relatively small variation in each cluster. Furthermore, cursive script and block print may have different mode of context variation. Clustering in Step 1 cannot be carried out appropriately for data like this, which spoils the accuracy improvement.

III. Partial Tied-Mixture and its Optimization

A. Partial Tied-Mixture

Now we propose Partial Tied-Mixture (PTM) model. The method selects M Gaussians from G Gaussians ($1 \leq M \leq G$) to built a GMM.

First examine the relationship between PTM and the conventional one. Fig.1(c) can be seen as special case of G=4, M=2 in PTM. The original Gaussians are four: A, B, C, and D. GMM of "a-r" consists of C and D. Similarly, GMMs of "i-r", "u-r", "e-r", and "o-r" consist of (A,B), (C,D), (A,B), and (C,D), respectively. That is, each GMM consists of two Gaussians selected from (A,B,C,D). However, arbitrary pair of Gaussians cannot be selected by the conventional method. For example, in Fig.1(c) the combinations (A,B) and (C,D) only exist. In this case, (A,C) cannot be used. Therefore, PTM can be seen as the generalization of the conventional one.

Next, let us see that PTM can deal with those cases as shown in Fig.2. It seems necessary in the case to partition the context in different way for cursive script and block print respectively, since the context varies in different ways in the two writing style. In Fig.2(b), cursive script is partitioned into three clusters (a-r,c-r), (b-r,d-r), and (e-r), while block print remains a single cluster. Each Gaussian E, F, G, and H is the estimation result of of each cluster, respectively. "a-r" is estimated by GMM consisting of F and H. "b-r", "c-r", "d-r", and "e-r" are estimated by GMM consisting of (E,H), (F,H), (E,H), and (G,H), respectively. Therefore, Fig.2(b) corresponds to the case of G=4, M=2 in PTM.

As shown above, Partial Tied-Mixture can deal with those cases with mixed writing style as Fig.2(b), as well as easier cases. For this improvement, conventional process does not work, in which context clustering and GMM estimation are carried out sequentially and separately. Based on the above consideration, we propose a method to carry out context clustering and GMM estimation concurrently. EM algorithm is employed for optimization. First, formulation for the case of M=1 is given in III-B. In this case, context is partitioned into mutually exclusive clusters, which can be seen as context clustering. Next, the method is generalized for the case of $M \geq 2$ in III-C. It can be seen as concurrent optimization of context clustering and GMM estimation.

Here, let us check the relation between PTM and tied-mixture. When M=G, the framework of PTM is the same as tied-mixture. In this case, estimation is difficult since contexts are overlapping like E, F, and G in Fig.2(b). The reason is the approximation of contexts by Gaussians do not likely to reflect the difference of the contexts, by using the whole set of the Gaussians for every contexts. On the other hand, the proposed method can deal with the matter by choosing the $M < G$ appropriate subset of Gaussians from the whole set. Here, higher recognition rate can be expected compared to tied-mixture.

B. Context Clustering by EM algorithm

Clustering of raw data can be accomplished by allocating each data to one of the Gaussian of the GMM estimated by

using EM algorithm. Estimating GMM is equivalent to determining the parameters a_m, μ_m, Σ_m so that the likelihood of all the data $\mathbf{X}$ be maximized, assuming a model in which the data are emitted by the m-th Gaussian $\mathcal{N}(\mathbf{x}; \mu_m, \Sigma_m)$, chosen with a priori probability a_m, where μ_m is the mean, Σ_m is the covariance matrix. Fig.3(a) is the conceptual diagram of the process.

Now let us adopt this approach to the clustering of context. Let $\mathbf{x}_{li}$ be a data which belongs to the context l ($l = 1, ..., L$), where $i = 1, ..., N_l$, N_l is the data size of context l. In the model of Fig.3(a), this time, the Gaussian m is chosen with a priori probability a_m for each context, not for each data, and all the data $\mathbf{x}_{l1}, ..., \mathbf{x}_{lN_l}$ belonging to the context l are emitted from the same Gaussian. Clustering of context can be realized by this model. Parameters in the model are determined so that the likelihood $p(\mathbf{X}|\Theta)$ is maximized, where $\mathbf{X}$ denotes all the data in all the context and Θ denotes all the parameters. This maximization can be solved by EM algorithm. The update formula can be derived in a similar way as data-wise clustering by GMM, as described below.

Let latent variable y_l be the index number of the Gaussian which the context l has chosen. Then, $y_l \in \{1, ..., G\}$. Let $\mathbf{Y} = (y_1, ..., y_L)$ be all the latent variables. EM algorithm increases the likelihood $p(\mathbf{X}|\Theta)$ by replacing Θ repeatedly by $\widehat{\Theta}$ which maximizes the Q-function defined below.

$$Q = \sum_{\mathbf{Y}} p(\mathbf{Y}|\mathbf{X}, \Theta) \log p(\mathbf{X}, \mathbf{Y}|\widehat{\Theta}) \tag{1}$$

By expanding Eq.(1), you get

$$Q = \sum_{l=1}^{L}\sum_{m=1}^{G} Y_{lm} \log \widehat{a}_m + \sum_{l=1}^{L}\sum_{m=1}^{G}\sum_{i=1}^{N_l} Y_{lm} \log \mathcal{N}(\mathbf{x}_{li}; \widehat{\mu}_m, \widehat{\Sigma}_m) \tag{2}$$

where $\widehat{a}_m, \widehat{\mu}_m, \widehat{\Sigma}_m$ are the parameters after update. Y_{lm} is a posteriori probability $P(y_l = m|\mathbf{X}, \Theta)$ that the context l chooses Gaussian m, which is given by

$$Y_{lm} = \frac{a_m \prod_{i=1}^{N_l} \mathcal{N}(\mathbf{x}_{li}; \mu_m, \Sigma_m)}{\sum_{m'=1}^{G} a_{m'} \prod_{i=1}^{N_l} \mathcal{N}(\mathbf{x}_{li}; \mu_{m'}, \Sigma_{m'})}. \tag{3}$$

Since the constraint is $\sum_{m=1}^{G} \widehat{a}_m = 1$, by using Lagrange's multiplier, the problem is to maximize $J = Q - \lambda(\sum_{m=1}^{G} \widehat{a}_m - 1)$, or to solve $\frac{\partial J}{\partial \widehat{\Theta}} = 0$. As a result, $\widehat{a}_m, \widehat{\mu}_m, \widehat{\Sigma}_m$ are given by

$$\widehat{a}_m = \frac{\sum_{l=1}^{L} Y_{lm}}{L}, \quad \widehat{\mu}_m = \frac{\sum_{l=1}^{L}\sum_{i=1}^{N_l} Y_{lm}\mathbf{x}_{li}}{\sum_{l=1}^{L}\sum_{i=1}^{N_l} Y_{lm}} \tag{4}$$

$$\widehat{\Sigma}_m = \frac{\sum_{l=1}^{L}\sum_{i=1}^{N_l} Y_{lm}(\mathbf{x}_{li} - \widehat{\mu}_m)(\mathbf{x}_{li} - \widehat{\mu}_m)^T}{\sum_{l=1}^{L}\sum_{i=1}^{N_l} Y_{lm}}. \tag{5}$$

The denominator of $\widehat{\mu}_m, \widehat{\Sigma}_m$ can also be expressed as $\sum_l N_l Y_{lm}$. Here, Eq.(3) is the E-step, Eq.(4)(5) is the M-step. Applying these two steps alternatively will increase the likelihood.

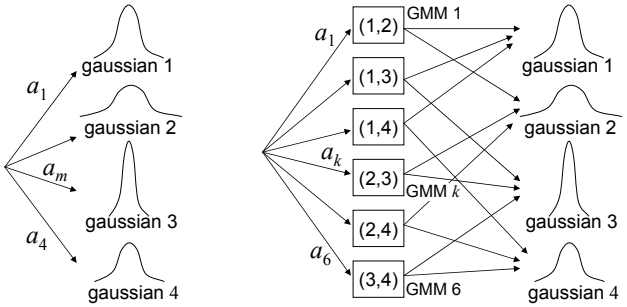

Figure 3. (a) A conceptual diagram of GMM (b) A conceptual diagram of PTM

Context clustering can be accomplished by allocating the context l to the Gaussian which maximizes the a posteriori probability Y_{lm} after the learning process.

C. Concurrent Optimization of Context Clustering & GMM

In this section, the methodology in III-B will be enhanced to treat the case of $M \geq 2$. The number of combination to select M Gaussians out of G to build a GMM is ${}_GC_M$. Assume that each context l selects k-th GMM with a priori probability a_k. Also assume that each data $\mathbf{x}_{li}$ of the context l selects a Gaussian of the k-th GMM independently. Fig.3(b) shows an example of G=4, M=2. In this case, ${}_4C_2$=6 GMMs are built. All the parameters Θ which maximize the likelihood $p(\mathbf{X}|\Theta)$ are estimated by EM algorithm as described below.

Let K be the number of GMM, then K=${}_GC_M$. Let C_k be the set of index numbers of Gaussians which constitute GMM k. For example, C_6=$\{3, 4\}$ for "GMM 6" in Fig.3(b). Let b_{km} be the mixing coefficient of Gaussian m in GMM k. GMM k is given by $\sum_{m \in C_k} b_{km} \mathcal{N}(\mathbf{x}; \mu_m, \Sigma_m)$. Here, we introduce a latent variable z_l, or the index number of the GMM which the context l selects. $z_l \in \{1, ..., K\}$. We also introduce a latent variable v_{li}, or the index number of the Gaussian which data $\mathbf{x}_{li}$ selects, then $v_{li} \in C_{z_l}$. Let the whole latent variable be denoted by $\mathbf{Z}$. In this case, $\mathbf{Z}$ includes z_l $(\forall l)$ and v_{li} $(\forall l, i)$. For other variables, we follow the naming convention in III-B.

By expanding Q-function, we get

$$Q = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta) \log p(\mathbf{X}, \mathbf{Z}|\widehat{\Theta}) \tag{6}$$

$$= \sum_{l=1}^{L}\sum_{k=1}^{K} Z_{lk} \log \widehat{a}_k + \sum_{l=1}^{L}\sum_{k=1}^{K}\sum_{i=1}^{N_l}\sum_{m \in C_k} Z_{lk} V_{likm} \log \widehat{b}_{km}$$
$$+ \sum_{l=1}^{L}\sum_{k=1}^{K}\sum_{i=1}^{N_l}\sum_{m \in C_k} Z_{lk} V_{likm} \log \mathcal{N}(\mathbf{x}_{li}; \widehat{\mu}_m, \widehat{\Sigma}_m), \tag{7}$$

where $\widehat{a}_k, \widehat{b}_{km}, \widehat{\mu}_m, \widehat{\Sigma}_m$ are parameters after update. Z_{lk} is the a posteriori probability of the context l to select GMM k, or $P(z_l = k|\mathbf{X}, \Theta)$, and V_{likm} is the a posteriori probability of the data $\mathbf{x}_{li}$ to select Gaussian m on the condition that

the context l selects GMM k, or $P(v_{li}=m|z_l=k,\mathbf{X},\Theta)$, and calculated as follows:

$$Z_{lk}=\frac{a_k\prod_{i=1}^{N_l}\sum_{m\in C_k}b_{km}\mathcal{N}(\mathbf{x}_{li};\mu_m,\Sigma_m)}{\sum_{k'=1}^{K}\left\{a_{k'}\prod_{i=1}^{N_l}\sum_{m\in C_{k'}}b_{k'm}\mathcal{N}(\mathbf{x}_{li};\mu_m,\Sigma_m)\right\}} \quad (8)$$

$$V_{likm}=\frac{b_{km}\mathcal{N}(\mathbf{x}_{li};\mu_m,\Sigma_m)}{\sum_{m'\in C_k}b_{km'}\mathcal{N}(\mathbf{x}_{li};\mu_{m'},\Sigma_{m'})}. \quad (9)$$

Since the constraints are $\sum_{k=1}^{K}\widehat{a}_k = 1$ and $\sum_{m\in C_k}\widehat{b}_{km}=1$ $(\forall k)$, by using Lagrange's multiplier, the problem is to maximize

$$J=Q-\lambda_0\left(\sum_{k=1}^{K}\widehat{a}_k-1\right)-\sum_{k=1}^{K}\lambda_k\left(\sum_{m\in C_k}\widehat{b}_{km}-1\right), \quad (10)$$

or to solve $\frac{\partial J}{\partial\widehat{\Theta}}=0$. Therefore in a similar fashion as in III-B, $\widehat{\mu}_m,\widehat{\Sigma}_m$ are solved in a form of a weighted mean and a weighted covariance matrix as:

$$\widehat{\mu}_m=\frac{\sum_{l=1}^{L}\sum_{i=1}^{N_l}\sum_{k=1}^{K}Z_{lk}V_{likm}\mathbf{x}_{li}}{\sum_{l=1}^{L}\sum_{i=1}^{N_l}\sum_{k=1}^{K}Z_{lk}V_{likm}} \quad (11)$$

$$\widehat{\Sigma}_m=\frac{\sum_{l=1}^{L}\sum_{i=1}^{N_l}\sum_{k=1}^{K}Z_{lk}V_{likm}(\mathbf{x}_{li}-\widehat{\mu}_m)(\mathbf{x}_{li}-\widehat{\mu}_m)^T}{\sum_{l=1}^{L}\sum_{i=1}^{N_l}\sum_{k=1}^{K}Z_{lk}V_{likm}} \quad (12)$$

$\widehat{a}_k,\widehat{b}_{km}$ are solved as:

$$\widehat{a}_k=\frac{\sum_{l=1}^{L}Z_{lk}}{L},\quad \widehat{b}_{km}=\frac{\sum_{l=1}^{L}\sum_{i=1}^{N_l}Z_{lk}V_{likm}}{\sum_{l=1}^{L}N_lZ_{lk}} \quad (13)$$

Eq.(8)(9) corresponds to E-step and Eq.(11)-(13) corresponds to M-step.

After learning phase has finished, for the output probability distribution of the context l, $p_l(\mathbf{x})$, although we may just choose the GMM k which maximizes the a posteriori probability Z_{lk}, here we compute predictive distribution [8] to refine estimation, since we know the a posteriori probability Z_{lk} of all the GMM. For mixing coefficient, b_{km} is not appropriate since it is the value shared by all the contexts. The value for each context l can be computed, like ordinary GMM estimation, as:

$$w_{lkm}=\frac{1}{N_l}\sum_{i=1}^{N_l}V_{likm}. \quad (14)$$

As a result, $p_l(\mathbf{x})$ is given by:

$$p_l(\mathbf{x})=\sum_{k=1}^{K}Z_{lk}\sum_{m\in C_k}w_{lkm}\mathcal{N}(\mathbf{x};\mu_m,\Sigma_m). \quad (15)$$

IV. Experiments

We conducted experiments on handwritten word recognition using American city name in CEDAR database [9]. We used 3,213 word images for training and 352 for evaluation, excluding those images with underlined words, geometrically mixed adjacent text lines, etc. from original datasets [1]. We used the slant adjustment technique by Ding et al. [10]. We applied neither size nor skew normalization. We used LGH feature [11]. Characters were handled with case-sensitivity. We employed 10 state left-to-right model. The maximum lexicon size was made to include 1,000 words, by adding training data to evaluation data. The lexicon size was varied from 10 to 1,000. We used dynamic lexicon technique in which lexicon was so arranged to include the correct one [2].

We built up three models for comparison:

CIM: Context-independent model. The number of mixture components was varied from 1 to 12.

STM [3]: Context-dependent tied-mixture model in which tying performed per the state position. Out of ten states, five is dependent on the preceding character, and the rest is dependent on the succeeding [3]. CIM's parameters are used for STM's initial values.

PTM: Proposed context-dependent Partial Tied-Mixture model. The same context configuration as STM. The same initial parameters as STM.

Fig.4 shows the error rate of CIM when lexicon size and the number of Gaussians are varied. The minimum error by lexicon size is marked by bold, also shown in the bottom line indicated by "min". Fig.5 shows the error rate, error reduction rate (ERR), and p-value in STM. ERR is the reduction rate of the error rate E compared to E' in CIM, and can be calculated as $ERR=\frac{E'-E}{E'}$. "min" value was used for E', E. The p-values were computed for all the pairs of G' and G'', where G' and G'' are the number of Gaussian in STM and CIM, respectively. The maximum values for G'' are shown in Fig.5 [4]. Fig.6 shows the error rate, ERR, and p-value in PTM when M=2. Two values of ERR are shown, compared to the case of CIM and STM. Likewise, two p-values are shown, compared to CIM and STM. The p-values less than 5% are written in bold style.

CIM>STM>PTM holds for every lexicon size when "min" values are compared. The effect of a context-dependent model can be seen in STM, but is more distinct in PTM. ERR compared to CIM is 10.1% to 28.8% for PTM, much higher than 3.0% to 6.0% for STM. ERR of PTM

[1]We used BD and BS datasets, which are the only randomly sampled ones. The two datasets are divided into 3,670 for training and 377 for evaluation by CEDAR.

[2]The real experiments were carried out for lexicon size 1,000 and the recognition rate for other lexicon sizes were computed by the data. If the rank of the true word is $1\leq r\leq 1000$ among 1,000 words in the lexicon, the probability of the event that the rank of the true word becomes 1st is $\frac{1000-rC_{s-1}}{999C_{s-1}}$ when the lexicon size is reduced to s, and was averaged over the evaluation data. This is equivalent to conducting all the s experiments. We calculated the error rate in two places of decimals for those cases with lexicon size up to 200, since the precision is virtually raised by the experimental process.

[3]This configuration is independent from the specific knowledge about the recognition object. This configuration is valid for arbitrary object.

[4]Wilcoxon signed-rank test was performed on the difference of true word's rank for lexicon size 1,000.

# of gaussian	lexicon size 10	20	50	100	200	500	1000
1	3.08	4.75	7.41	10.0	13.4	19.3	24.4
2	2.14	3.46	5.55	7.58	10.3	15.2	19.6
3	***1.92***	3.15	5.16	7.05	9.59	14.7	20.2
4	2.07	3.29	5.16	6.69	***8.47***	***12.0***	***15.9***
5	2.12	3.33	5.23	6.87	8.79	12.4	***15.9***
6	***1.92***	2.99	4.83	6.55	8.58	12.2	16.2
7	1.93	***2.96***	***4.73***	***6.44***	8.51	12.2	16.2
8	2.01	3.05	4.81	6.50	8.67	12.6	16.8
9	2.00	3.02	4.86	6.66	8.95	13.0	16.8
10	2.00	3.00	4.74	6.47	8.75	13.0	17.3
11	2.09	3.11	4.83	6.50	8.70	12.9	17.0
12	2.12	3.11	4.85	6.56	8.76	12.6	16.2
min	1.92	2.96	4.73	6.44	8.47	12.0	15.9

Figure 4. Error rate of CIM

# of gaussian	lexicon size 10	20	50	100	200	500	1000	p-value CIM
5	2.01	3.22	5.12	6.72	8.52	11.9	***15.1***	79.8
6	1.84	2.94	4.80	6.50	8.36	***11.5***	***15.1***	24.4
7	***1.80***	2.84	4.59	***6.25***	***8.16***	***11.5***	***15.1***	6.4
8	1.85	2.91	4.69	6.37	8.47	12.4	16.5	39.6
9	1.91	2.91	4.65	6.33	8.41	12.0	15.9	52.6
10	1.83	***2.82***	***4.54***	***6.25***	8.52	12.8	17.0	62.8
11	1.95	2.95	4.63	6.26	8.42	12.4	16.5	60.8
12	1.98	2.96	4.67	6.39	8.60	12.5	16.5	83.2
min	1.80	2.82	4.54	6.25	8.16	11.5	15.1	
ERR CIM	6.0	4.8	4.1	3.0	3.7	3.7	5.4	

Figure 5. Error rate, ERR, p-value in STM

directly compared to STM is 6.7% to 24.2%, showing the efficiency.

The minimum p-value of STM compared to CIM is 6.4% when G'=7, which does not necessarily mean difference, under 5% significance level. On the other hand, p-value of PTM when G=10, is 0.4% and 1.1%, compared to CIM and STM, respectively, meaning significantly lower error rate.

The error rate of PTM is lower than that of STM for every lexicon size, when G and G' are set to 7, which gives the maximum performance for STM. PTM seems more precise than, and as efficient as tied-mixture, since computational efficiency depends on the number of Gaussians.

V. Conclusion

In this paper, a novel parameter tying method for a context-dependent HMM called "Partial Tied-Mixture" has been proposed. Conventional context clustering does not seem effective for handwriting recognition, since there exist distinctively different letter shapes in the same category. We proposed a method in which M Gaussians are selected from G Gaussians and are optimized by EM algorithm. We first formulated a context clustering by means of the EM algorithm, and generalized it to the above mentioned optimization method. Experiments on handwritten word recognition indicated up to 24.2% error reduction compared to the tied-mixture method. We have also confirmed that the error rate is lower than the conventional method, assuming that the computation time is the same.

Future work includes shorter learning time. Learning time increases exponentially with G and M. The optimal G and M increase and the learning becomes more and more difficult, as the amount of learning data increases. It may be solved by pruning those branches with small values.

# of gaussian	lexicon size 10	20	50	100	200	500	1000	p-value CIM	STM
5	1.68	2.76	4.55	6.13	8.00	11.5	15.6	9.5	28.1
6	1.59	2.74	4.72	6.46	8.45	11.9	15.9	13.7	35.8
7	1.55	2.60	4.42	6.13	8.13	11.4	14.2	***4.9***	16.7
8	1.41	2.34	4.07	5.77	7.68	***10.3***	***12.8***	***0.9***	***3.4***
9	1.44	2.33	4.06	5.87	7.96	11.2	14.2	***1.3***	6.5
10	***1.36***	***2.24***	***3.90***	***5.60***	***7.61***	10.7	13.6	***0.4***	***1.1***
11	1.45	2.33	4.02	5.78	7.92	11.5	15.1	***1.2***	***2.8***
12	1.59	2.50	4.21	6.02	8.40	12.5	16.8	25.2	48.9
min	1.36	2.24	3.90	5.60	7.61	10.3	12.8		
ERR CIM	28.8	24.5	17.6	12.9	10.1	13.4	19.6		
ERR STM	24.2	20.7	14.1	10.3	6.7	10.1	15.1		

Figure 6. Error rate, ERR, p-value in PTM

References

[1] A. Kosmala, J. Rottland, and G. Rigoll, "Improved on-line handwriting recognition using context dependent hidden markov models," *Proc. 4th ICDAR*, vol. 2, pp. 641–644, Aug. 1997.

[2] J. Tokuno, N. Inami, S. Matsuda, M. Nakai, H. Shimodaira, and S. Sagayama, "Context-dependent substroke model for HMM-based on-line handwriting recognition," *Proc. 8th IWFHR*, pp. 78–83, Aug. 2002.

[3] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, and K. Subramanian, "Multi-lingual offline handwriting recognition using hidden markov models: A script-independent approach," *Springer Book Chapter on Arabic and Chinese Handwriting Recognition*, vol. 4768, pp. 231–250, Mar. 2008.

[4] M. Schussler and H. Niemann, "A HMM-based system for recognition of handwritten address words," *Proc. 6th IWFHR*, pp. 505–514, Aug. 1998.

[5] G. A. Fink and T. Plotz, "On the use of context-dependent modeling units for HMM-based offline handwriting recognition," *Proc. 9th ICDAR*, pp. 729–733, Sep. 2007.

[6] R. El-Hajj, C. Mokbel, and L.L-Sulem, "Recognition of Arabic handwritten words using contextual character models," *Proc. SPIE*, vol. 6815, pp. 681 503–681 503–9, Jan. 2008.

[7] A.-L. Bianne, C. Kermorvant, and L. L.-Sulem, "Context-dependent HMM modeling using tree-based clustering for the recognition of handwritten words," *Proc. SPIE*, vol. 7534, p. 75340I, Jan. 2010.

[8] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[9] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. PAMI*, vol. 16, no. 5, pp. 550–554, May 1994.

[10] Y. Ding, F. Kimura, Y. Miyake, and M. Shridhar, "Accuracy improvement of slant estimation for handwritten words," *Proc. 15th ICPR*, vol. 4, pp. 527–530, Sep. 2000.

[11] J. A. Rodriguez and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," *Proc. 1st ICFHR*, pp. 7–12, Aug. 2008.