HMM-Based Recognition of Online Handwritten Mathematical Symbols Using Segmental K-means Initialization and A Modified Pen-up/down Feature

Lei Hu
Department of Computer Science
Rochester Institute of Technology
Rochester, USA
lei.hu@rit.edu

Richard Zanibbi
Department of Computer Science
Rochester Institute of Technology
Rochester, USA
rlaz@cs.rit.edu

***Abstract*—This paper presents a recognition system based on Hidden Markov Model (HMM) for isolated online handwritten mathematical symbols. We design a continuous left to right HMM for each symbol class and use four online local features, including a new feature: normalized distance to stroke edge. A variant of segmental K-means is used to get initialization of the Gaussian Mixture Models' parameters which represent the observation probability distribution of the HMMs. The system obtains top-1 recognition rate of 82.9% and top-5 recognition rate of 97.8% on a dataset containing 20281 training samples and 2202 testing samples of 93 classes of symbols. For multi-stroke symbols, the top-1 recognition rate is 74.7% and the top-5 recognition rate is 95.5%. For single-stroke symbols, the top-1 recognition rate is 86.8% and the top-5 recognition rate is 98.9%. (MacLean et al., 2010) applied dynamic time warping algorithm on all the 70 classes of single-stroke symbols. Their top-1 recognition rate is 85.8%, and top-5 recognition rate is 97.0%. Our system gets top-1 recognition rate of 85.5% and top-5 recognition rate of 99.1% on the same 70 classes of single-stroke symbols.**

***Keywords*-Hidden Markov Model; mathematical symbol recognition; segmental K-means**

I. Introduction

Mathematical expressions are an indispensable component of scientific and technical literatures [1]. So far the most popular way to enter mathematical expressions is either in a linear format (e.g., TEX), or by using a structured editor (e.g., equation editor available with MS-Word) [2]. Producing large and complicated expressions in these two ways requires a lot of time and mental effort. With the emergence of pen-based electronic devices, such as PDAs and tablet PCs, people can simply write mathematical expressions on the electronic tablet to let the computer recognize them automatically.

Recognition of mathematical expressions includes two major steps: symbol recognition and structural analysis [1]. Symbol recognition is the basis of the structural analysis. It consists of two phases: symbol segmentation and isolated symbol recognition. The input data of online handwritten mathematical expression is a set of strokes, and a mathematical symbol may comprise more than one stroke. Symbol segmentation aims to transform the sequence of strokes into a set of symbols, which will be classified in the isolated symbol recognition stage.

In this paper, we will focus on the recognition of isolated online handwritten mathematical symbols based on Hidden Markov Model (HMM). We establish a continuous left to right HMM for each symbol class. A variant of segmental K-means is used to get initialization of the Gaussian Mixture Models' parameters representing the observation probability distribution of the HMMs. We modify pen-up/down information into a new feature, normalized distance to stroke edge. We use four online local features in total, which contain more information about each point. Experiment results show that the variant of segmental K-means can produce better initialization of the Gaussian Mixture Models' parameters and normalized distance to stroke edge is a better feature than the pen-up/down information.

II. Related Work

A number of approaches have been proposed for online handwritten mathematical symbol recognition. A group of methods is based on nearest neighbor scheme. Smithies et al. [3] proposed a fast user-trained algorithm based on nearest neighbor classification in a feature space of approximately 50 dimensions. Vuong et al. [4] proposed an extended elastic matching algorithm. Elastic matching is achieved through calculating the minimum distance between the template symbol and input symbol with dynamic programming. During the matching process, every point of the input symbol is matched against that in the template symbol. Apart from Euclidean distance between points, the extended elastic matching algorithm also considers slope and curvature information during its matching process. MacLean et al. [5] presented a greedy approximate solution to the dynamic time warping algorithm for recognizing single-stroke symbols and the time complexity of the algorithm is linear.

There have also been many rule-based methods. Fitzgerald et al. [6] used fuzzy logic to extract features, such as Line, C-shape and O-shape, and classify symbols. In symbol recognition phase, the system uses two types of fuzzy rules:

high-level rules and low-level rules. High-level rules define the properties the input symbol must have if it belongs to a particular class. Low-level rules assess the extent to which these properties are present. Belaid et al. [7] proposed an approach based on decision tree classifier. The non-leaf nodes in the decision tree are the set of rules to classify the input symbol. Curvature, direction and drawing length are used as features.

Another group of methods combine different classifiers. Garain et al. [2] presented a symbol recognition system to combine two different kinds of classifier. The first classifier employs a nearest neighbor classification and the second one uses a left to right HMM. Both the classifiers make use of direction change and trajectory length as features. The system combines the two classifiers through three ways: highest rank method, Borda count and logistic regression, and logistic regression gets the best performance.

There are several mathematical symbol recognition systems based on statistical approach. Matsakis [8] presented a symbol recognition method based on a quadratic discriminant classifier. Winkler et al. [9] proposed a symbol recognition system based on HMM. Their system extracts both on-line features and off-line features. They builds three semi-continuous left to right HMMs for each symbol to combine the classification results. Two HMMs use the off-line features while one HMM uses the on-line features. The online features they used are the local position, the sine and cosine value of the angle between the horizontal axis and the vector connecting the previous and the current point, and the information whether the current point belongs to a stroke or to an interpolated hidden stroke.

III. METHODOLOGY

A. preprocessing

Our preprocessing method is similar to the one in [10], but has fewer steps. Our preprocessing procedure just consists of four steps: duplicate point filtering, size normalization, smoothing and resampling. These steps reduce noise and unuseful information for classification.

Duplicate point filtering: duplicate point is the point that has the same (x, y) coordinates as the previous point and cannot give any useful information for classification.

Size normalization: the class of a symbol is independent of its size, therefore size normalization is needed to eliminate the variation of size. It is achieved by transforming the y coordinate's range to be $[0, 1]$ while preserving the width-height aspect ratio.

Smoothing: smoothing is used to reduce the noise information caused by the digital pen's jitter. Except the first point and the last of every stroke, the other points' coordinates are replaced by the average of the coordinates of current point, the previous point and the following point.

Resampling: the original points are recorded equidistantly in time but not in space. In order to remove the influence of writing velocity, we resample each symbol to 30 points along the original trajectory with equal distance between the consecutive points by the method in [11].

B. feature extraction

Liwicki et al. [12] applied a sequential forward search on a feature set in order to discover which features are significant for handwriting recognition. A Hidden Markov Model and a bidirectional long short-term memory network (BLSTM) based recognizer were used as recognition engines. They applied many operations to reduce noise and normalize the skew, slant, width and height before feature extraction.

There are 25 features in the feature set: (1) pen-up/down, (2) hat-feature, (3) speed, (4) normalized x-coordinate, (5) normalized y-coordinate, (6,7) cosine and sine of writing direction, (8,9) cosine and sine of curvature, (10-18) context map, (19) vicinity aspect, (20) vicinity curliness, (21) vicinity linearity, (22,23) cosine and sine of vicinity slope, (24) ascenders, and (25) descenders.

The experiment results [12] show that the recognition rate with only five features approaches the recognition rate using all the 25 features. The experiment results [12] also show that the first five iterations of the sequential forward search algorithm with HMM based classifier and BLSTM based classifier have selected the same best five features. In addition, in the first five iterations of the sequential forward search algorithm with HMM based classifier, the ranking of the first five features does not change. It can be concluded that the five features are very stable and contain more important information than other features for the classification. The best five features are the cosine of the slope, the normalized y-coordinate, the density in the center of the context maps, the pen-up/down information, and the sine of the curvature.

We use all four online features among the best five features: the cosine of the slope, the normalized y-coordinate, the pen-up/down information, and the sine of the curvature. A 4-dimensional feature vector is computed for each point of the sample. Because the number of the features is small, we can get a more efficient classifier, in terms of computation and storage.

Pen-up/down: a binary feature denoting whether the digital pen has contact with the electronic tablet or not at time t.

Normalized distance to stroke edge (NDTSE): in order to add the location information to the pen-up/down feature, we take the distances to the beginning and the end of the stroke into account and replace the pen-up/down feature with NDTSE. The new feature can be computed as :

$$NDTSE(s,t) = \begin{cases} 1 - \frac{|d_e - d_b|}{l_s}, & \text{for actual stroke} \\ -(1 - \frac{|d_e - d_b|}{l_s}), & \text{for interpolated stroke,} \end{cases} \quad (1)$$

where l_s represents the length of the stroke s which the current point $x(t), y(t)$ belongs to; d_e represents the distance between the current point and the last point of s; d_b represents the distance between the current point and the first point of s. Actual stroke is the visible stroke, while interpolated stroke is the hidden parts of the trajectory, where the digital pen does not contact with the electronic tablet. For the point belongs to actual stroke, NDTSE is nonnegative; for the point belongs to interpolated stroke, NDTSE is nonpositive. Fig. 1 visualizes the new feature.

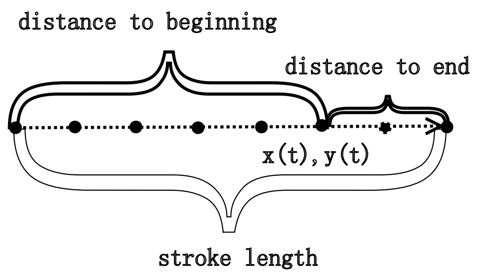

Figure 1. New feature: normalized distance to stroke edge, containing the pen-up/down information and the location information of the current point

Normalized y-coordinate: the vertical position after size normalization.

Vicinity slope α: the vicinity slope of the current point $(x(t), y(t))$ is represented by the cosine and sine of the angle between the straight line connecting the point $(x(t-2), y(t-2))$ and the point $(x(t+2), y(t+2))$ and the horizontal across the point $(x(t-2), y(t-2))$. α in Fig. 2 represents the slope.

Curvature β: the curvature of the current point $(x(t), y(t))$ is represented by cosine and sine of the angle between the straight line joining point $(x(t-2), y(t-2))$ and point $(x(t), y(t))$ and the straight line joining point $(x(t), y(t))$ and point $(x(t+2), y(t+2))$. β in Fig. 2 represents the curvature.

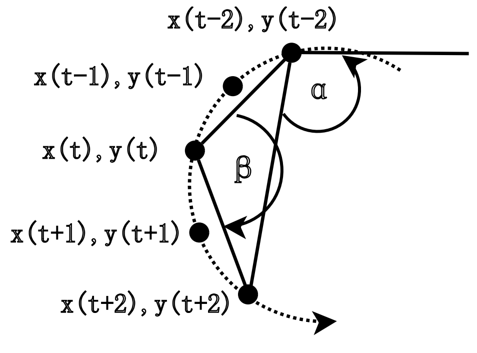

Figure 2. Slope(α) and curvature(β)

C. HMM classifier

An HMM process is a doubly stochastic process [13]. The underlying process is hidden from observation and is represented by a state transition probability matrix, where the current state just depends on the previous state. The observable process is determined by the underlying process and is represented by an observation probability distribution function, where the current observation just depends on the current state. So an HMM is specified by the parameter set (A, B, π). A denotes the state transition probability matrix; B denotes the observation probability distribution; π is the initial state distribution.

Each written symbol can be represented by a sequence of feature vectors O, defined as

$$O = O_1, O_2, \cdots, O_T, \tag{2}$$

where O_t is the feature vector observed at time t. The goal of the HMM classifier is to find the probability that a specific class is the most likely to occur given a sequence of observations. Therefore the symbol recognition problem is to compute

$$\operatorname*{argmax}_i P(\lambda_i \mid O), \tag{3}$$

where λ_i is the i'th symbol class. Bayes' Rule gives

$$P(\lambda_i \mid O) = \frac{P(O \mid \lambda_i)P(\lambda_i)}{P(O)}. \tag{4}$$

$P(O)$ is the same for all classes. If all classes have the same priori probability $P(\lambda_i)$, then the symbol recognition can be regarded as that of computing

$$i^* = \operatorname*{argmax}_{1 \le i \le N} P(O \mid \lambda_i). \tag{5}$$

1) Model Selection: There is no theoretically optimal method to choose the type of model (ergodic or left to right), the model size (number of states) and observation probability distribution (discrete or continuous, single or multi-mixture) [13] for an HMM. The type of model, the model size and the observation probability distribution are determined empirically.

In our HMMs, we use the linear topology. For each state, only the transition to itself or the next state is permitted. The observation probability for a given feature vector is determined by Gaussian Mixture Models and the covariance matrix of the mixture component is diagonal.

To choose the model size and the number of Gaussian components per state, we did experiments on the ten digits extracted from the corpus of handwritten mathematical expressions [14] to find the effect of number of states and number of Gaussians on the recognition rate. The experiment results are shown in Table I. The recognition rate is the average of ten trials.

We assigned the same weight to the top-1 and top-5 recognition rate. It can be found with 4 Gaussians per state, model with 6 states acquires the best performance. When the number of states is fixed to be 6, model with 5 Gaussians

Table I
AVERAGE TEST RECOGNITION RATE OF TEN TRIALS ON TEN DIGITS EXTRACTED FROM THE CORPUS [14] OF DIFFERENT COMBINATIONS OF STATES AND GAUSSIANS

Model Size	top-1	top-5
3 states 4 Gaussians	0.960	0.997
4 states 4 Gaussians	0.965	0.998
5 states 4 Gaussians	0.966	0.998
6 states 4 Gaussians	0.968	0.999
7 states 4 Gaussians	0.969	0.994
6 states 2 Gaussians	0.960	0.997
6 states 3 Gaussians	0.963	0.995
6 states 5 Gaussians	0.974	0.999
6 states 6 Gaussians	0.968	0.994

per state gets the best performance. Therefore, each model has six states and each state contains five Gaussians in our system.

2) Initialization: Theoretically speaking, the re-estimation process of Baum-Welch algorithm can assign values to the HMM's parameters which can make the likelihood function to get a local maximum [13]. But there is no straightforward way to choose good initial estimates of the HMM parameters to guarantee that the local maximum is the global maximum or a strong local maximum of the likelihood function.

In most cases, either random or uniform initial estimates of the initial state distribution π and state transition probability matrix A is enough with Baum-Welch algorithm for producing useful reestimates of these parameters. But the reestimates of the Gaussian parameters are very sensitive to the initial estimates [15]. Therefore good initial estimates of Gaussian parameters are necessary. In this paper, we set the initial state distribution to be $\pi = 1, 0, 0, 0, 0, 0$ and keep it fixed during the training process. That means the first feature vector out of a sequence is fixed to the first state. But we don't fix the last feature vector out of a sequence to the last state. Discrete uniform distribution is used to give the initial state transition probability matrix A.

We use a variant of segmental K-means algorithm [15] to get the initial parameters of observation probability distribution B. We first assign random values to the Gaussian parameters. Then over five iterations, we use the Viterbi algorithm [16][17] to get the optimal path, having to terminate at the final state, of all observation sequences and segment the feature vector of each point according to the optimal path into the six states. After each state gets the set of the feature vectors that are assigned to it in the current and all previous iterations, K-means algorithm is used to cluster the observations into five clusters and update the Gaussian parameters of each state. But segmental K-means segments all training sequence according to the optimal path given by the Viterbi algorithm. Each state just can get the observations that occur within it in the current iteration.

Table II
93 SYMBOL CLASSES CAN BE RECOGNIZED BY OUR SYSTEM

0	1	2	3	4	5	6	7	8	9
a	b	c	d	e	f	g	h	i	j
k	l	m	n	o	p	q	r	s	t
u	v	w	x	y	z	A	B	C	D
E	F	G	H	I	J	K	L	M	N
O	P	Q	R	S	T	U	V	W	X
Y	Z	α	β	δ	Δ	ϵ	=	γ	Γ
$\geq$	$>$	-	∞	$\int$	[	(	μ	ϕ	π
Π	+	ψ	]	ρ	)	σ	Σ	$\surd$	τ
θ	ξ	ζ							

3) Training: There are a number of methods for the HMM training, such as Baum-Welch algorithm [18], Genetic Algorithm [19], maximum margin learning [20] and maximum mutual information estimation [21]. In our system, we use the Baum-Welch algorithm. The Baum-Welch algorithm is a type of EM(expectation-maximization) method based on the maximum likelihood criterion. After each iteration of the Baum-Welch algorithm, $P(O \mid \bar{\lambda}) > P(O \mid \lambda)$ and $\bar{\lambda}$ will replace λ, where $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ represents the re-estimated model and $\lambda = (A, B, \pi)$ denotes the previous one. The algorithm will run until it is convergent or the maximum iteration is finished.

4) Recognition: In the recognition phase, all HMMs are used with the Forward algorithm [13] to calculate the probability of the observation sequence, $O = O_1 O_2 \cdots O_T$, given the model λ, $P(O \mid \lambda)$. The symbol with the maximal class conditional probability will be selected as the class label.

IV. DATASET AND EXPERIMENT RESULTS

We extract all symbols from a new publicly available, ground-truthed corpus of handwritten mathematical expressions [14], getting 100 different symbol classes. These symbols were written by 20 writers. We discard six symbol classes whose samples are less than 50 and the symbol 'dot', because they are not adequate or not suitable for training the corresponding HMMs. Table II shows all the 93 classes of symbols which can be recognized in our system. The dataset is unbalanced, and different symbol have different numbers of samples. For each class of symbol, we use 90% samples as the training set and the other 10% as the testing set. There are 20281 samples in the training set and 2202 samples in the testing set.

We did experiments to find out whether segmental K-means can get better initialization of the Gaussian parameters. The control group use K-means to get the parameters for the 30 Gaussian components and assign them to six states randomly. Fig. 3 shows the comparison of the average recognition rate of 20 trials and the standard deviation between using K-means and segmental K-means. With segmental K-means, the average recognition rates are higher, and the standard deviations are much lower. Two-tailed, unequal variance t-test (n=20) shows the increase of top-5 recognition rate for all symbols is statistically significant, when $\alpha = 0.05$. It can be concluded that segmental K-means can give better initialization of the Gaussian parameters.

With segmental K-means, we did experiments to compare the performance using NDTSE with using pen-up/down. Fig. 4 shows the comparison of the average recognition rate of 20 trials and the standard deviation between using pen-up/down and NDTSE. The recognition rates with NDTSE are higher. Two-tailed, unequal variance t-test (n=20) shows the increases of top-1 and top-5 recognition rates for all symbols are both statistically significant, when $\alpha = 0.05$. It can be concluded that NDTSE is a better feature than pen-up/down.

With segmental K-means and NDTSE, we did experiments to compare the performance using different prior probability represented by the symbol's sample ratio with using the same prior probability. With different prior probability, the system has slightly higher recognition rates, but the improvements are not statistically significant. This shows our system is robust and does not rely on the prior knowledge of the specific data set.

Using segmental K-means, NDTSE and different prior probability, our system obtains best top-1 recognition rate of 82.9% among 20 trials for all symbols. In the trial producing the best top-1 accuracy for all symbols, top-5 recognition rate for all symbols is 97.8%; for multi-stroke symbols, the top-1 recognition rate is 74.7% and the top-5 recognition rate is 95.5%; for single-stroke symbols, the top-1 recognition rate is 86.8% and the top-5 recognition rate is 98.9%.

MacLean et al. [5] applied their method on all the single-stroke symbols of the same corpus, 70 classes in total, including 0-4, 6-9, a-e, g-h, k-s, u-w, y-z, B-D, G, L-O, Q-S, U-W, Z, α, β, δ, Δ, ϵ, γ, $>$, -, ∞, $\int$, [, (, $<$, μ, Ω, Π,], ρ,), σ, Σ, $\sqrt{}$, θ, ξ and ζ. The best top-1 recognition rate is 85.8%, and the best top-5 recognition rate is 97.0%. We also applied our system to the 70 kinds of symbols, getting best top-1 recognition rate of 85.5%. In the trial producing the best top-1 accuracy, the top-5 recognition rate is 99.1%.

Through analyzing the confusion matrix, we find many classification errors are caused by symbols are classified to the classes with the similar shape, such as the number '0', the capital letter 'O' and the small letter 'o'. This explains the differences between the top-1 recognition rates and the top-5 recognition rates. Therefore the recognition rate can be improved by building discriminatory classifiers aiming for these confused classes or collapsing these confused classes to a single class.

V. Conclusion

This paper presents a system for recognition of isolated online handwritten mathematical symbols. The classifier is based on HMM and we use 4-dimensional online local features. A new feature, normalized distance to stroke edge, is defined based on pen-up/down information. A variant of segmental K-means is used to get initialization of the Gaussian Mixture Models' parameters which represent the observation probability distribution of the HMMs. The initial experiment results are encouraging in light of we just use four features.

For future work, more extensive experiments should be conducted for performance comparison with different preprocessing methods and different feature sets. In addition, we will optimize the topology of the HMMs and extend the research on recognition of isolated mathematical symbols to the recognition of mathematical expressions.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1016815. The authors would like to thank Dr. Pengcheng Shi for his valuable comments.

References

[1] K.-F. Chan and D.-Y. Yeung, "Mathematical expression recognition: a survey," *International Journal on Document Analysis and Recognition*, vol. 3, no. 1, pp. 3–15, Aug. 2000.

[2] U. Garain and B. Chaudhuri, "Recognition of online handwritten mathematical expressions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 6, pp. 2366–2376, 2004.

[3] S. Smithies, K. Novins, and J. Arvo, "A handwriting-based equation editor," *Proc. Graphics Interface*, pp. 84–91, June 1999.

[4] B. Vuong, Y. He, and S. Hui, "Towards a web-based progressive handwriting recognition environment for mathematical problem solving," *Expert Systems with Applications*, vol. 37, no. 1, pp. 886–893, 2010.

[5] S. Maclean and G. Labahn, "Elastic matching in linear time and constant space," in *Proc. Ninth IAPR Int'l. Workshop on Document Analysis Systems*. ACM, 2010, pp. 551–554.

[6] J. Fitzgerald, F. Geiselbrechtinger, and T. Kechadi, "Mathpad: A fuzzy logic-based recognition system for handwritten mathematics," *In Proc. International Conf. on Document Analysis and Recognition*, vol. 2, pp. 694–698, 2007.

[7] A. Belaid and J.-P. Haton, "A syntactic approach for handwritten mathematical formula recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, pp. 105–111, Jan. 1984.

[8] N. Matsakis, "Recognition of handwritten mathematical expressions," Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, May 1999.

[9] H.-J. Winkler, "HMM-based handwritten symbol recognition using on-line and off-line features," *In Proc. International Conference on Acoustics, Speech, and Signal Processing*, pp. 3438–3441, 1996.

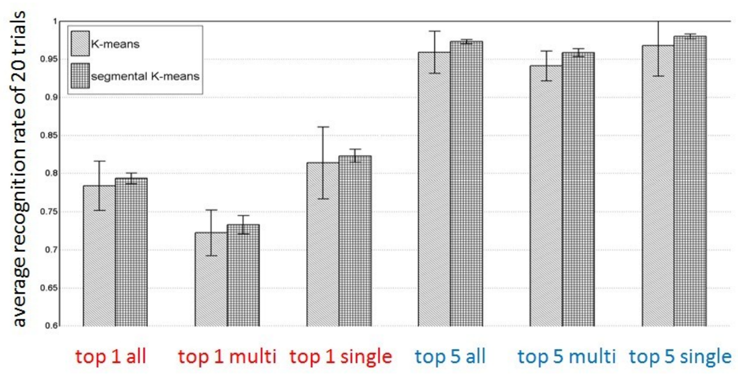

Figure 3. Comparison of average recognition rate and standard deviation between using K-means and segmental K-means

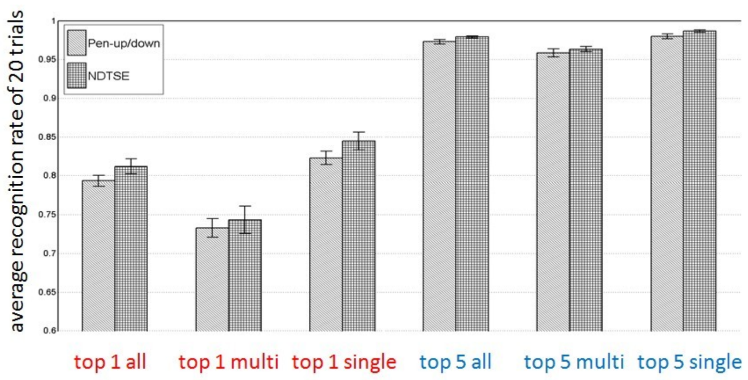

Figure 4. Comparison of average recognition rate and standard deviation between using Pen-up/down and NDTSE

[10] Y. Zhang, G. Shi, and J. Yang, "HMM-based online recognition of handwritten chemical symbols," *In Proc. International Conference on Document Analysis and Recognition*, pp. 1255–1259, 2009.

[11] M. Pastor, A. Toselli, and E. Vidal, "Writing speed normalization for on-line handwritten text recognition," *In Proc. International Conference on Document Analysis and Recognition*, pp. 1–5, 2005.

[12] M. Liwicki and H. Bunke, "Feature selection for HMM and BLSTM based handwriting recognition of whiteboard notes," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 5, pp. 907–923, 2009.

[13] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *In Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[14] S. MacLean, G. Labahn, E. Lank, M. Marzouk, and D. Tausky, "Grammar-based techniques for creating ground-truthed sketch corpora," *International Journal on Document Analysis and Recognition*, pp. 1–21, May 2010.

[15] L. R. Rabiner, B.-H. Juang, S. E. Levinson, and M. M. Sondhi, "Recognition of isolated digits using hidden markov models with continuous mixture densities," *AT&T technical journal*, vol. 64, no. 6, pp. 1211–1234, 1985.

[16] J. Forney, G.D., "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268 – 278, 1973.

[17] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260 – 269, Apr. 1967.

[18] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, vol. 41, no. 1, pp. 164–171, 1970.

[19] C. Chau, S. Kwong, C. Diu, and W. Fahrner, "Optimization of HMM by a genetic algorithm," *In Proc. International Conference on Acoustics, Speech, and Signal Processing*, pp. 1727–1730, 1997.

[20] T.-M.-T. Do and T. Artieres, "Maximum margin training of Gaussian HMMs for handwriting recognition," *In Proc. International Conference on Document Analysis and Recognition*, pp. 976–980, 2009.

[21] P. C. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 25–47, Jan. 2002.