2011 International Conference on Document Analysis and Recognition

Convolutional Neural Network Committees For Handwritten Character Classification

Dan Claudiu Cireşan and Ueli Meier and Luca Maria Gambardella and Jürgen Schmidhuber
IDSIA
USI, SUPSI
6928 Manno-Lugano, Switzerland

{*dan,ueli,luca,juergen*}*@idsia.ch*

***Abstract*—In 2010, after many years of stagnation, the MNIST handwriting recognition benchmark record dropped from 0.40% error rate to 0.35%. Here we report 0.27% for a committee of seven deep CNNs trained on graphics cards, narrowing the gap to human performance. We also apply the same architecture to NIST SD 19, a more challenging dataset including lower and upper case letters. A committee of seven CNNs obtains the best results published so far for both NIST digits and NIST letters. The robustness of our method is verified by analyzing 78125 different 7-net committees.**

***Keywords*-Convolutional Neural Networks; Graphics Processing Unit; Handwritten Character Recognition; Committee**

I. Introduction

Current automatic handwriting recognition algorithms are not bad at learning to recognize handwritten characters. Convolutional Neural Networks (CNNs) [1], [2] are among the most suitable architectures for this task. Recent CNN work focused on computer vision problems such as recognition of 3D objects, natural images and traffic signs [3]–[5], image denoising [6] and image segmentation [7]. Convolutional architectures also seem to benefit unsupervised learning algorithms applied to image data [8], [9]. Reference [10] reported an error rate of 0.4% on the MNIST handwritten character recognition dataset [2], using a fairly simple CNN, plus elastic training image deformations to increase the training data size. In 2010, using graphics cards (GPUs) to greatly speed up training of plain but deep Multilayer Perceptrons (MLPs), an error rate of 0.35% was obtained [11]. Such an MLP has many more free parameters than a CNN. Here we report experiments using CNNs trained on MNIST as well as on the more challenging NIST SD 19 database [12], which contains 482,925 training and 82,587 test characters (i.e. upper- and lower-case letters as well as digits). On GPUs, CNNs can be successfully trained on such extensive databases within reasonable time ($\approx$ 1 to 6 hours of training, depending on the task).

At some stage in the classifier design process one usually has collected a set of reasonable classifiers. Typically one of them yields best performance. Intriguingly, however, the sets of patterns misclassified by different classifiers do not necessarily greatly overlap. Here we focus on improving recognition rates using committees of neural networks. Our goal is to produce a group of classifiers whose errors on various parts of the training set differ as much as possible. We show that for handwritten digit recognition this can be achieved by training identical classifiers on data preprocessed/normalized in different ways [13]: 0.31% error rate for a committee of simple, big and deep MLPs on MNIST. Other approaches aiming at optimally combining neural networks [14], [15] do not do this, thus facing the problem of strongly correlated individual predictors. Furthermore, we simply average individual committee member outputs, instead of optimizing their combinations [15], [16], which would cost additional valuable training data.

II. Training the individual nets

CNNs are used as base classifiers [4]. The same architecture is used for experiments on NIST SD 19 and MNIST. The nets have an input layer of 29×29 neurons followed by a convolution layer with 20 maps of 26×26 neurons and filters of size 4×4. The next hidden layer is a max-pooling layer [17], [18] with a 2×2 kernel which has its outputs connected to another convolution layer containing 40 maps of 9×9 neurons each. The last max-pooling layer is reducing the map size to 3×3 by using filters of size 3×3. A fully connected layer of 150 neurons is connected to the max-pooling layer. The output layer has one neuron per class, e.g. 62 for NIST SD 19 and 10 for MNIST.

All CNNs are trained in full online mode with an annealed learning rate and continually deformed data—the images from the training set are distorted at the beginning of every epoch. The following elastic deformation parameters $\sigma = 6$ and $\alpha = 36$ are used together with independent horizontal and vertical scaling of at most 15% and at most $\pm 15^\circ$ of rotation for all experiments [11]. Deformations are essential to prevent overfitting, and greatly improve generalization.

GPUs accelerate the deformation routine by a factor of 10 (only elastic deformations are GPU-optimized), and the network training procedure by a factor of 60 [4]. We pick the

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.229

trained CNN with the lowest validation error, and evaluate it on the corresponding test set.

III. Forming a committee

We perform experiments on the original and six preprocessed datasets. Preprocessing is motivated by writing style variations resulting in different aspect ratios of the handwritten characters. Prior to training, we therefore normalize the width of all characters to 10, 12, 14, 16, 18 and 20 pixels, except for characters in {1,i,l,I} and in the original data [13].

The training procedure of a network is summarized in Figure 1a. Each network is trained separately on normalized or original data. The normalization is done for all digits in the training set prior to training (normalization stage). During each training epoch every single character is distorted in a different way. The committees are formed by simply averaging the corresponding outputs as shown in Figure 1b.

For each of the preprocessed or original datasets, five differently initialized CNNs are trained for the same number of epochs. This allows for performing an error analysis of the outputs of the $5^7 = 78125$ possible committees of seven nets, each trained on one of the seven datasets. We report mean and standard deviation as well as minimum and maximum recognition rate.

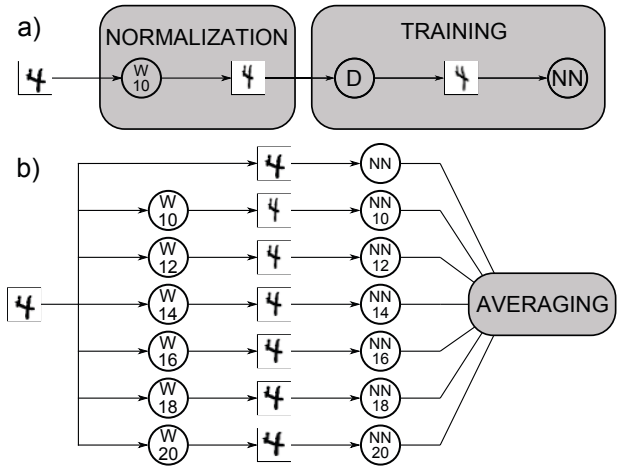

Figure 1. *a) Training a committee member:* Original training data (left digit) is normalized (W10) prior to training (middle digit). The normalized data is distorted (D) for each training epoch (right digit) and fed to the neural network (NN). Each depicted digit represents the whole training set. *b) Testing with a committee:* If required, the input digits are width-normalized (W blocks) and then processed by the corresponding NNs. A committee averages the outputs of its CNNs.

IV. Experiments

We use a system with a Core i7-920 (2.66GHz), 12 GB DDR3 and four graphics cards: 2 x GTX 480 and 2 x GTX 580. Details of our GPU implementation are explained in [4], [11].

Our method is applied to two handwritten character datasets: subsets from NIST SD 19 and digits from MNIST (Table I).

We use the proven learning rate schedule of our previous work [4], [5], [11], [13]: in every epoch (up to a predetermined number of epochs) we multiply the learning rate (initially 0.001) by a factor of 0.993 until it reaches 0.00003.

Table I
Datasets.

Name	Type	Training set	Test set	#Classes
MNIST	digits	60000	10000	10
NIST SD 19	digits&letters	482925	82587	62
NIST SD 19	digits	344307	58646	10
NIST SD 19	letters	138618	23941	52
NIST SD 19	merged	138618	23941	37
NIST SD 19	lowercase	69096	12000	26
NIST SD 19	uppercase	69522	11941	26

A. Experiments with NIST Special Database 19

NIST SD 19 contains over 800,000 handwritten characters. We follow the recommendations of the authors and build standard training and test sets. The 128×128 character images are uncompressed; their bounding-boxes are resized to 20×20. The resulting characters are centered within a 29×29 image. This normalizes all the characters in the same way MNIST digits are already normalized.

1) Digits & letters: We train five differently initialized nets on each preprocessed dataset as well as on the original data, for a total of 35 CNNs (Table II). Each CNN is trained for 30 epochs by on-line gradient descent. The number of epochs is limited due to the size of NIST SD 19: training a single net for the 62 class problem takes almost six hours.

Table II
Test error rate [%] of the 35 CNNs trained on NIST SD 19, 62 class task. Wxx - width of the character is normalized to xx pixels.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	14.72	14.12	13.72	13.55	13.77	13.82	14.32
2	14.73	14.21	13.80	14.20	13.93	13.04	14.73
3	13.92	14.12	13.50	13.57	13.81	14.15	14.57
4	14.07	14.42	13.46	13.47	13.76	13.63	14.05
5	14.14	13.69	13.91	13.92	13.50	13.60	13.72
	Committees						
	Average 11.88±0.09			Min 11.68		Max 12.12	

The average committee is significantly better than any of the individual CNNs. Even the worst committee is better than the best net. Recognition errors of around 12% may still seem large, but one should consider that most errors stem from confusions of classes that look very similar: {0,o,O}, {1,l,i,I}, {6,G}, {9,g}, and all confusions between similar uppercase and lowercase letters (see below). Without any additional contextual information, it is literally impossible to distinguish those.

We therefore also train various nets on digits, all letters, a merged letter set, and also on lowercase and uppercase letters separately. This drastically decreases the number of confused classes and also makes it possible to compare our results to other published results. We are not aware of any previous study publishing results on the challenging full 62 class problem.

2) Digits: Table III summarizes the results of 35 identical CNNs trained for 30 epochs on digits.

Table III
TEST ERROR RATE [%] OF THE 35 CNNS TRAINED ON NIST SD 19 DIGITS. WXX - WIDTH OF THE CHARACTER IS NORMALIZED TO XX PIXELS.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	1.37	1.42	1.22	1.14	1.22	1.12	1.34
2	1.40	1.35	1.35	1.21	1.12	1.13	1.40
3	1.52	1.28	1.24	1.19	1.16	1.15	1.36
4	1.47	1.35	1.43	1.29	1.24	1.21	1.49
5	1.63	1.39	1.39	1.25	1.16	1.27	1.42
	Committees						
	Average 0.81±0.02		Min 0.73		Max 0.91		

Our average error rate of 0.81% on digits compares favorably to other published results, 1.88% [19], 2.4% [20] and 3.71% [21]. Again, as for the 62 class problem, the committees significantly outperform the individual nets.

3) Letters: Table (IV) summarizes the results of 35 identical CNNs trained for 30 epochs on letters. Again the same architecture is used.

Table IV
TEST ERROR RATE [%] OF THE 35 CNNS TRAINED ON NIST SD 19 LETTERS. WXX - WIDTH OF THE CHARACTER IS NORMALIZED TO XX PIXELS.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	24.69	24.76	24.12	24.50	23.98	23.01	24.49
2	25.53	24.97	24.08	24.06	23.52	23.35	24.93
3	25.09	25.14	24.06	24.28	24.20	23.58	24.62
4	25.27	24.74	24.53	24.59	24.51	23.87	24.45
5	25.65	25.91	24.74	25.12	24.39	23.69	25.38
	Committees						
	Average 21.41±0.16		Min 20.80		Max 22.13		

Class boundaries of letters in general and uppercase and lowercase letters in particular are separated less clearly than those of digits. However, many obvious error types are avoidable by different experimental set-ups, i.e., by ignoring case, merging classes, and considering uppercase and lowercase classes independently. Ignoring case, average error is three times smaller (7.58%).

4) Merged letters: Table V summarizes results of 35 identical CNNs trained for 30 epochs on merged letters. Uppercase and lowercase letters in {C,I,J,K,L,M,O,P,S,U,V,W,X,Y,Z} are merged as suggested in the NIST SD 19 documentation [12], resulting in 37 distinct classes for this task.

Ignoring case for only 15 out of 26 letters suffices to avoid most case confusions. Ignoring case completely reduces error only slightly, under loss of ability to distinguish the case of the 11 remaining letters.

Table V
TEST ERROR RATE [%] OF 35 CNNS TRAINED ON NIST SD 19 MERGED LETTERS. WXX - WIDTH OF THE CHARACTER IS NORMALIZED TO XX PIXELS.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	10.40	10.11	9.54	9.67	9.30	9.38	10.01
2	10.38	10.20	9.99	9.47	9.68	9.34	10.25
3	10.69	10.23	9.50	9.52	9.55	9.87	10.08
4	11.10	10.21	10.20	9.95	9.86	9.76	10.21
5	10.87	10.80	10.35	9.46	9.71	10.03	10.64
	Committees						
	Average 8.21±0.11		Min 7.83		Max 8.56		

5) Upper- or lowercase letters: Further simplifying the task by considering uppercase and lowercase letters independently yields even lower error rates (Tables VI, VII).

Table VI
TEST ERROR RATE [%] OF 35 CNNS TRAINED ON NIST SD 19 UPPERCASE LETTERS. WXX - WIDTH OF THE CHARACTER IS NORMALIZED TO XX PIXELS.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	3.08	2.90	2.80	2.51	2.60	2.55	2.79
2	3.03	2.73	2.84	2.70	2.78	2.53	2.70
3	3.33	2.96	2.83	2.65	2.84	2.65	2.68
4	3.29	3.22	2.96	2.65	2.65	2.60	2.87
5	3.23	2.97	2.70	2.78	2.86	2.64	2.70
	Committees						
	Average 1.91±0.06		Min 1.71		Max 2.15		

Uppercase letters are much easier to classify than lowercase letters—error rates are four times smaller. Shapes of uppercase letters are better defined, and in-class variability due to different writing styles is generally smaller.

Table VII
TEST ERROR RATE [%] OF 35 CNNS TRAINED ON NIST SD 19 LOWERCASE LETTERS. WXX - WIDTH OF THE CHARACTER IS NORMALIZED TO XX PIXELS.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	10.22	9.30	9.42	9.25	9.08	8.87	9.44
2	10.30	9.75	9.91	9.63	9.19	9.32	8.84
3	10.51	9.88	9.95	9.18	8.88	9.82	9.29
4	10.12	9.73	10.39	9.63	9.05	10.05	10.04
5	10.82	9.56	10.08	9.53	9.58	9.59	10.24
	Committees						
	Average 7.71±0.14		Min 7.16		Max 8.28		

B. Experiments on MNIST

The MNIST data is already preprocessed such that the width or height of each digit is 20 pixels. Our CNNs are trained for around 800 epochs, with small improvement after 500 epochs. Training one net takes almost 14 hours.

The average error rate of $0.27 \pm 0.02\%$ is by far the best result published on this benchmark. In Figure 2 all 69 errors of all committees are shown, together with the true labels and the majority votes of the committees. Digits are sorted

Table VIII
TEST ERROR RATE [%] OF THE 35 CNNS TRAINED ON MNIST. WXX - WIDTH OF THE CHARACTER IS NORMALIZED TO XX PIXELS.

Trial	W10	W12	W14	W16	W18	W20	ORIG
1	0.49	0.39	0.40	0.40	0.39	0.36	0.52
2	0.48	0.45	0.45	0.39	0.50	0.41	0.44
3	0.59	0.51	0.41	0.41	0.38	0.43	0.40
4	0.55	0.44	0.42	0.43	0.39	0.50	0.53
5	0.51	0.39	0.48	0.40	0.36	0.29	0.46
	Committees						
	Average 0.27±0.02			Min 0.17		Max 0.37	

in descending order of how many committees committed the same error, indicated as percentages at the bottom of each digit. The first six errors were committed by all committees—obviously the corresponding digits are either wrongly labeled or very ambiguous, and the majority vote seems correct. Each committee fails to recognize between 17 to 37 digits out of the 69 presented errors.

Figure 2. The 69 errors of all committees, the label (up left), the committee majority vote (up right), and the percentage of committees committing a particular error (down left).

C. Summary of experiments

Table IX summarizes our results and compares to previously published results where available. For letters it was difficult to find any publications reporting results for similar experimental set-ups. To the best of our knowledge, our results are far better (30-350%) than any published result.

Error rates for digits are significantly lower than those for letters. Training nets with case-insensitive letter labels makes error rates drop considerably, indicating that most errors of nets trained on 52 lowercase and uppercase letters are due to confusions between similar classes. A generic letter recognizer should therefore be trained on a merged letter dataset. If required, case conflicts have to be resolved a posteriori, using additional (e.g, contextual) information.

All experiments use the same net architecture and deformation parameters, which are not fine-tuned to increase classification accuracy. We rely on committees to improve recognition rates. Additional tests, however, show that our deformation parameters are too big for small letters—using 20% lower values decreases error rates by another 1.5%.

Table IX
AVERAGE ERROR RATES OF COMMITTEES FOR ALL THE EXPERIMENTS, ± ONE STANDARD DEVIATION [%], PLUS RESULTS FROM THE LITERATURE.*CASE INSENSITIVE

Data	Committee	Published results		
MNIST	**0.27**±0.02	0.40 [10]	0.35 [11]	0.31 [13]
NIST:				
all (62)	**11.88**±0.09			
digits (10)	**0.81**±0.02	5.06 [22]	3.71 [21]	1.88 [19]
letters (52)	**21.41**±0.16	30.91 [23]		
letters* (26)	**7.58**±0.09	13.00 [24]	13.66 [23]	
merged (37)	**8.21**±0.11			
uppercase (26)	**1.91**±0.06	10.00 [24]	6.44 [25]	11.51 [23]
lowercase (26)	**7.71**±0.14	16.00 [24]	13.27 [23]	

For commercial OCR, recognition speed is of great interest. Our nets check almost 10000 characters per second. At first glance, a committee of seven such nets is seven times slower than a single net, but we can run the nets in parallel on seven different GPUs, thus keeping the committee throughput at the single net level.

We won the ICDAR 2011 Offline Chinese Character Recognition Competition [26] with a CNN architecture virtually identical to the one of this paper, all differences being dictated by the input size of the nets. The good results on this huge problem with 3755 classes further confirm the flexibility, robustness and scalability of our algorithm.

V. CONCLUSION

Simple training data pre-processing gave us experts with errors less correlated than those of different nets trained on the same or bootstrapped data. Hence committees that simply average the expert outputs considerably improve recognition rates. Our committee-based classifiers of isolated handwritten characters are the first on par with human performance [27], [28], and can be used as basic building blocks of any OCR system (all our results were achieved by software running on powerful yet cheap gaming cards).

ACKNOWLEDGMENT

This work was partially supported by Swiss CTI, Commission for Technology and Innovation, Project n. 9688.1 IFF: Intelligent Fill in Form and by a FP7-ICT-2009-6 EU Grant under Project Code 270247: A Neuro-dynamic Framework for Cognitive Robotics: Scene Representations, Behavioral Sequences, and Learning.

REFERENCES

[1] K. Fukushima, "Neocognitron: A self-organizing neural network for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, November 1998.

[3] F.-J. Huang and Y. LeCun, "Large-scale learning with svm and convolutional nets for generic object categorization," in *Proc. Computer Vision and Pattern Recognition Conference (CVPR'06)*. IEEE Press, 2006.

[4] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *International Joint Conference on Artificial Intelligence*, 2011, to appear.

[5] D. C. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in *International Joint Conference on Neural Networks*, 2011, to appear.

[6] V. Jain and H. S. Seung, "Natural image denoising with convolutional networks," in *Advances in Neural Information Processing Systems (NIPS 2008)*, 2008.

[7] S. C. Turaga, J. F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H. S. Seung, "Convolutional networks can learn to generate affinity graphs for image segmentation," *Neural Computation*, vol. 22, pp. 511–538, 2010.

[8] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th International Conference on Machine Learning*, 2009, pp. 609–616.

[9] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional Networks," in *Proc. Computer Vision and Pattern Recognition Conference (CVPR 2010)*, 2010.

[10] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Seventh International Conference on Document Analysis and Recognition*, 2003, pp. 958–963.

[11] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep big simple neural nets for handwritten digit recognition," *Neural Computation*, vol. 22, no. 12, pp. 3207–3220, 2010.

[12] P. J. Grother, "NIST special database 19 - Handprinted forms and characters database," National Institute of Standards and Thechnology (NIST), Tech. Rep., 1995.

[13] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Handwritten digit recognition with a committee of deep neural nets on gpus," Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Tech. Rep. IDSIA-03-11, 2011.

[14] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[15] N. Ueda, "Optimal linear combination of neural networks for improving classification performance," *IEEE Trans. Pattern Analysis and Mach. Intelli.*, vol. 22, no. 2, pp. 207–215, 2000.

[16] S. Hashem, "Optimal linear combination of neural networks," *Neural Networks*, vol. 10, pp. 599–614, 1997.

[17] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. International Conference on Computer Vision*, 2009.

[18] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International Conference on Artificial Neural Networks*, 2010.

[19] J. Milgram, M. Cheriet, and R. Sabourin, "Estimating accurate multi-class probabilities with support vector machines," in *Int. Joint Conf. on Neural Networks*, 2005, pp. 1906–1911.

[20] L. Oliveira, R. Sabourin, F. Bortolozzi, and C. Suen, "Automatic recognition of handwritten numerical strings: a recognition and verification strategy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1438–1454, Nov. 2002.

[21] E. Granger, P. Henniges, and R. Sabourin, "Supervised Learning of Fuzzy ARTMAP Neural Networks Through Particle Swarm Optimization," *Pattern Recognition*, vol. 1, pp. 27–60, 2007.

[22] P. V. W. Radtke, R. Sabourin, and T. Wong, "Using the rrt algorithm to optimize classification systems for handwritten digits and letters," in *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 2008, pp. 1748–1752.

[23] A. L. Koerich and P. R. Kalva, "Unconstrained handwritten character recognition using metaclasses of characters," in *Intl. Conf. on Image Processing (ICIP)*, 2005, pp. 542–545.

[24] P. R. Cavalin, A. de Souza Britto Jr., F. Bortolozzi, R. Sabourin, and L. E. S. de Oliveira, "An implicit segmentation-based method for recognition of handwritten strings of characters." in *SAC'06*, 2006, pp. 836–840.

[25] E. M. Dos Santos, L. S. Oliveira, R. Sabourin, and P. Maupin, "Overfitting in the selection of classifier ensembles: a comparative study between pso and ga," in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. ACM, 2008, pp. 1423–1424.

[26] L. Cheng-Lin, Y. Fei, W. Qiu-Feng, and W. Da-Han, "ICDAR 2011 chinese handwriting recognition competition," in *Proc. International Conference on Document Analysis and Recognition (ICDAR)*, 2011, to appear.

[27] Y. LeCun, L. D. Jackel, L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Learning algorithms for classification: A comparison on handwritten digit recognition," in *Neural Networks: The Statistical Mechanics Perspective*, J. H. Oh, C. Kwon, and S. Cho, Eds. World Scientific, 1995, pp. 261–276.

[28] F. Kimura, N. Kayahara, Y. Miyake, and M. Shridhar, "Machine and human recognition of segmented characters from handwritten words," in *Int. Conf. on Document Analysis and Recognition*, 1997, pp. 866–869.