Adaptive Local Subspace Classifier in on-line recognition of handwritten characters

Jorma Laaksonen, Matti Aksela, Erkki Oja
Laboratory of Computer and Information Science,
Helsinki University of Technology,
P.O.BOX 5400, Fin-02015 HUT, Finland

Jari Kangas
Nokia Research Center
P.O. BOX 100, Fin-33721 Tampere, Finland
email: {jorma.laaksonen,matti.aksela,erkki.oja}@hut.fi, jari.kangas@research.nokia.com

Abstract

Subsystems for on-line recognition of handwriting are needed in personal digital assistants (PDAs) and other portable handheld devices. We have developed a recognition system which enhances its accuracy by applying continuous adaptation to the user's writing style. The forms of adaptation we have experimented with take place simultaneously with the normal operation of the system and, therefore, there is no need for separate training period of the device. The present implementation uses Dynamic Time Warping (DTW) in matching the input characters with the stored prototypes. The DTW algorithm implemented with Dynamic Programming (DP) is, however, both time and memory consuming. In our current research, we have experimented with methods that transform the elastic templates to pixel images which can then be recognized by using statistical or neural classification. The particular neural classifier we have used is the Local Subspace Classifier (LSC) of which we have developed an adaptive version.

Introduction

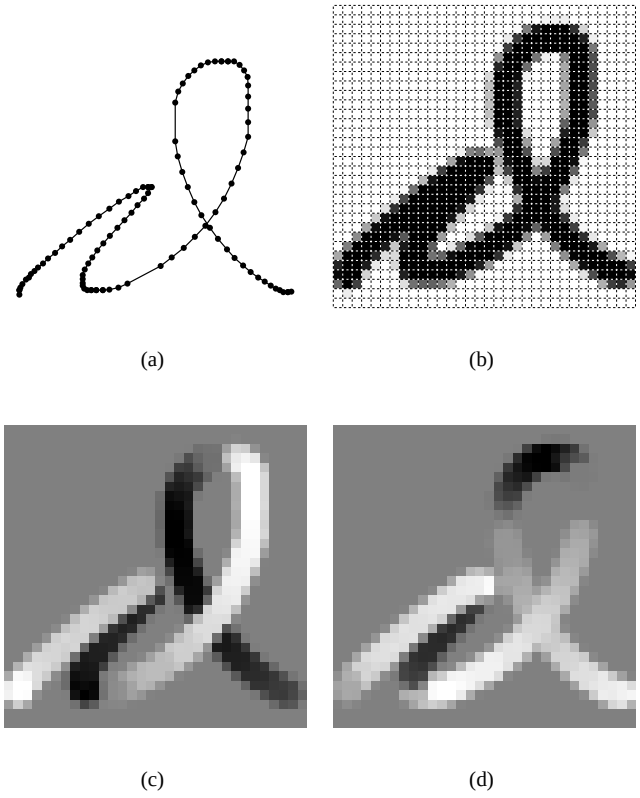
We have developed an on-line recognition system for handwritten characters which is aimed at being used in personal digital assistants (PDAs) and other portable handheld devices [3]. The system applies continuous adaptation to the user's writing style. The adaptation takes place simultaneously with the normal operation of the system and, therefore, there is no need for a separate training period of the device.

Character recognition in our system is currently performed by using Dynamic Time Warping (DTW) matching of elastic templates [6]. However, DTW algorithm implemented with Dynamic Programming (DP) is both time and memory consuming. Therefore, we have searched for alternative recognition methods. Also, the implementation of effective committee classifiers necessitates the use of different types of member classifiers whose errors should be as independent from each other as possible. In our current study, we transformed the deformable templates to pixel images. Feature vectors of constant dimension, and thus usable with statistical and neural classifiers, were then extracted from the images. The characters were recognized with methods which were earlier found to be effective for off-line recognition of numerals written on paper [2]. As the particular neural-type classification algorithm we have used an adaptive version of the Local Subspace Classifier (LSC) [1].

Feature extraction

The on-line recognition system for isolated characters produces xy-coordinate pairs and pressure information at a fixed frequency. In our Wacom ArtPad II equipment, the spatial resolution is 100 l/mm and the sampling frequency 200 Hz. Figure 1a displays a handwritten character 'd'. The sampled points are shown with dots connected by straight lines.

In our first feature extraction method, the straight lines connecting the measured xy-points were thickened to the width of $2r$ units in a coordinate system where the image was centered in a 1024×1024-sized frame. The thickening process was carried out by drawing filled circles of radius r along the path of the stylus. The original frame was then downsampled to the size of 32×32 by averaging. Figure 1b illustrates the character 'd' after downscaling the thickened



(a) (b)

(c) (d)

Figure 1: Handwritten 'd' in various forms: (a) original points connected with lines, (b) thickened image in 32×32 frame, (c) vertical direction image, and (d) horizontal direction image.

stroke. The particular value of $r = 50$ has been used. The effect of the averaging in downsampling can be observed as grey shades around the character boundary.

In the second feature extraction method, two 32×32-sized images were created instead of one. The directions of the lines connecting the sampled pen positions were used as additional information when creating the images. In the first one, illustrated in Figure 1c, the vertical component of the direction of pen movement was used in thickening the path. The filling value was obtained as $f_v = \sin\theta$ where θ is the line direction in polar coordinates. Likewise, the horizontal part $f_h = \cos\theta$ was used in the second image as depicted in Figure 1d. In both illustrations, white represents positive and black negative values, respectively.

The feature extraction process was in both cases continued by concatenating the pixel values of the grey-scale images. This gave rise to 1024-dimensional pattern vectors in the former and to 2048-dimensional vectors in the latter case. The available training data constituted of the total of 8461 lowercase characters and numerals written by 21 subjects. The covariance matrix of this set was calculated after the feature extraction. The first 64 eigenvectors of the covariance matrix were used in projecting the pattern vectors to a 64-dimensional feature space using the Karhunen-Loève Transform (KLT).

Local Subspace Classifier (LSC)

The Local Subspace Classifier (LSC) method [1] models the distribution of the pattern classes in a nonparametric fashion by using existing prototypes to span lower-dimensional local subspaces in the feature space. Instead of measuring distances to the discrete prototypes, as with the k-Nearest Neighbor (k-NN) classification rule, the distance is now defined between the input sample and the linear manifold nearest to it. The LSC procedure can be defined as follows. A D-dimensional linear manifold $\mathcal{L}$ of the d-dimensional real space is defined by a matrix $\mathbf{U} \in R^{d\times D}$ of rank D, and an offset vector $\boldsymbol{\mu} \in R^d$, provided $D \leq d$,

$$\mathcal{L}_{\mathbf{U},\boldsymbol{\mu}} = \{\mathbf{x} \mid \mathbf{x} = \mathbf{U}\mathbf{z} + \boldsymbol{\mu} \; ; \; \mathbf{z} \in R^D\} \; .$$

The same manifold can alternatively be defined by $D+1$ prototypes provided that the set of prototypes is not degenerate. The prototypes forming the classifier are marked $\mathbf{m}_{ij}$ where $j = 1, \ldots, c$ is the index of the class and $i = 1, \ldots, N_j$ indexes the prototypes in that class. When classifying a vector $\mathbf{x}$, the following is performed for each class $j = 1, \ldots, c$:

1. Find the $D+1$ prototypes closest to $\mathbf{x}$ and denote them $\mathbf{m}_{0j}, \ldots, \mathbf{m}_{Dj}$.
2. Form a $d \times D$-dimensional basis from the vectors $\{\mathbf{m}_{1j} - \mathbf{m}_{0j}, \ldots, \mathbf{m}_{Dj} - \mathbf{m}_{0j}\}$.
3. Orthonormalize the basis to obtain the matrix $\mathbf{U}_j = (\mathbf{u}_{1j} \; \ldots \; \mathbf{u}_{Dj})$.
4. Find the projection of $\mathbf{x} - \mathbf{m}_{0j}$ on the manifold $\mathcal{L}_{\mathbf{U}_j,\mathbf{m}_{0j}}$:
$$\widehat{\mathbf{x}}'_j = \mathbf{U}_j\mathbf{U}_j^T(\mathbf{x} - \mathbf{m}_{0j}) \; .$$
5. Calculate the residual of $\mathbf{x}$ relative to the manifold:
$$\widetilde{\mathbf{x}}_j = \mathbf{x} - (\mathbf{m}_{0j} + \widehat{\mathbf{x}}'_j) = (\mathbf{I} - \mathbf{U}_j\mathbf{U}_j^T)(\mathbf{x} - \mathbf{m}_{0j}) \; .$$

The vector $\mathbf{x}$ is then classified according to minimal $\|\widetilde{\mathbf{x}}_j\|$ to the class j, i.e., the classification decision function $g(\mathbf{x})$ can be written as

$$g_{\mathrm{LSC}}(\mathbf{x}) = \operatorname*{argmin}_{j=1,\ldots,c} \|\widetilde{\mathbf{x}}_j\| \; . \tag{1}$$

In any case, the residual length from the input vector $\mathbf{x}$ to the linear manifold is equal to or smaller than the distance

to the nearest prototype, i.e., $\|\widetilde{\mathbf{x}}_j\| \leq ||\mathbf{x} - \mathbf{m}_{0j}||$. The LSC method degenerates to the 1-NN rule when $D = 0$.

By introducing the multipliers $\{c_{0j}, \ldots, c_{Dj}\}$ forming the coefficient vector $\mathbf{c}_j = (c_{0j} \ \ldots \ c_{Dj})^T$, and using the matrix $\mathbf{M}_j = (\mathbf{m}_{0j} \ \ldots \ \mathbf{m}_{Dj})$, the projection vector $\widehat{\mathbf{x}}_j$ can be expressed

$$\widehat{\mathbf{x}}_j = \mathbf{m}_{0j} + \widehat{\mathbf{x}}'_j = \sum_{i=0}^{D} c_{ij}\mathbf{m}_{ij} = \mathbf{M}_j\mathbf{c}_j \ , \ \ \sum_{i=0}^{D} c_{ij} = 1 \ .$$

The explicit values for the coefficients c_{ij} can be solved with matrix pseudo-inversion

$$\mathbf{c}_j = \mathbf{M}_j^{\dagger}\widehat{\mathbf{x}}_j = (\mathbf{M}_j^T\mathbf{M}_j)^{-1}\mathbf{M}_j^T\widehat{\mathbf{x}}_j \ .$$

In a modification of the basic LSC method, named the Convex Local Subspace Classifier (LSC+), the c_{ij} coefficients are required to be non-negative. This is accomplished by setting the negative coefficients to zero, which equals to removing the corresponding vectors from the basis. The projection and the coefficients are then iteratively resolved until an orthogonal projection to the convex subspace spanned by a subset of the nearest prototypes has been found.

In the present experiments, we created the LSC classifiers individually and adaptively for each writer in the test set. During the adaptive creation of the prototype set used by the classifier, we utilized two distinct rules controlling the inclusion of the input character into the classifier. The 'E' rule stated that the prototype was added only if the LSC classifier had misclassified the input. The 'A' rule forced the addition of every input character.

Setup of Experiments

We have used a corpus of 21 writers in initializing the recognizer in a user-independent fashion. The training set of 8461 numerals and lowercase characters including three Scandinavian diacriticals 'ä', 'ö', and 'å', was first used in the determination of the KLT transformation matrix. Our initial experiments with the first feature extraction method revealed that the best recognition accuracy for k-NN classification was obtained with the dimensionality $d = 45$ for the feature vectors. This value was exclusively used in later studies. The same experiments showed that the best value for k in the k-NN classification rule was 1.

The initial user-independent 1-NN classifier was formed by using the K-means algorithm [4, 5] to extract a set of typical representatives for each character class. The value for K was varied in the experiments between 1 and 10, thus resulting to prototype set sizes between 39 and 390. The adaptation to each test writer's writing style was performed as follows. Every input character was classified with both the user-independent 1-NN classifier and the adaptive user-dependent LSC classifier. The joint classification decision of the two was given by the one with shorter distance to either to the nearest prototype or the nearest local subspace, respectively. This was possible as the both types of classifiers are based on the Euclidean distance metrics and measure the residuals in same units.

If the class provided by the 1-NN classifier was incorrect, the corresponding prototype in the K-means-initialized prototype set was removed. The input character was added to the LSC prototype set either always (the 'A' rule) or only if the LSC classifier had misclassified it (the 'E' rule). As a result, the size of the 1-NN classifier decreased while the size of the LSC classifier increased during the adaptation. As a consequence, the classification decisions were increasingly determined by the latter.

The user-dependent adaptive behavior of the system was evaluated with an independent set of 8 writers, each of whom had contributed approximately 580 lowercase and digit characters. Two figures of performance were recorded for every writer: first, the average error rate during the whole adaptation run, second, the error rate for the last 200 characters written. These figures are denoted in the tables as *total* and *final* error rates, respectively.

Results

The averages of the results for the individual writers when the first feature extraction method was used are collected in Table 1 whereas those for the second method are in Table 2. In both tables, all combinations of $K \in \{1, \ldots, 10\}$ and $D \in \{0, \ldots, 9\}$ were evaluated and the the one which yielded the lowest *final* error rate was regarded as the best result and recorded in the table. This was repeated six times, for all combinations of the three classifiers (1-NN, LSC, and LSC+) and the two inclusion rules ('E' and 'A').

	'E' rule				'A' rule			
	total	*final*	K	D	*total*	*final*	K	D
1-NN	22.0	19.0	7		16.1	11.2	7	
LSC	18.6	13.9	10	4	13.5	8.1	9	4
LSC+	18.5	13.8	7	5	13.7	8.1	9	6

Table 1: Average recognition error rate percentages for the first feature extraction method.

The classifier labeled 1-NN in the two tables was included for reference. These results were obtained by adding the input characters to the prototype set of the new classifier but instead of the LSC classification rule, the 1-NN rule was used. This resulted to an adaptive recognizer otherwise similar to LSC but without the linear manifolds spanned between the included prototypes. The results for the LSC

	'E' rule				'A' rule			
	total	*final*	K	D	*total*	*final*	K	D
1-NN	22.4	18.6	8		17.2	11.5	5	
LSC	18.9	13.9	10	3	14.4	8.3	10	8
LSC+	18.7	13.9	10	2	14.8	8.4	8	9

Table 2: Average recognition error rate percentages for the second feature extraction method.

classifiers are clearly better than those for the 1-NN classifier in every instance. This is in accordance with the earlier results [2] in which the LSC technique outperformed the k-NN classification and other neural and statistical methods used in the comparison.

Within the tables it can be seen that there are no significant differences between the LSC and LSC+ results. The former can therefore be preferred because it is computationally lighter. The adaptation rule 'A' seems to be always better than the 'E' rule. This is an indication that the adaptive behavior is the more effective the more prototypes are added. The adaptation of LSC is also possible to be implemented by modifying the existing prototypes instead of adding new ones if the available memory or computation resources are restrictive [2].

Between the tables it can be seen that the two feature extraction methods yield quite similar recognition rates. The additional information on the line directions in the latter features seems therefore not to be beneficial in classification. This result was quite unexpected as people tend to draw character strokes in a consistent order and direction, which should be helpful when designing a classifier.

The results in the tables can be compared to the non-adaptive recognition rates obtained with the user-independent 1-NN classifier only. For both feature sets, the approximate error rates were 40% *total* and 42% *final*. These rates were obtained with $K = 10$ prototypes per class. The fact that the *final* error rate in this case is higher than the *total* rate results from the writers being bored in their writing and producing in the end characters that are harder to classify than in the beginning. As the classifier was static, this phenomenon could not be compensated by adaptation.

More importantly, the present results can be compared to the previous results obtained with other recognition methods in our system. The *total* error rates for the Dynamical Time Warping classifier have been about half of those obtained in the present experiments [7]. The *final* error rates with DTW have been even less than half of those reported here. Still, the current results for adaptive classification outperform the recognition accuracies we have obtained for non-adaptive classification with the DTW method. Also, they are clearly better than the results we have obtained earlier by using matching of chain codes as the recognition method.

Conclusions

The present results of the performed experiments have showed that the proposed method is not as powerful recognition method for on-line characters as the elastic matching realized with Dynamical Time Warping. This can mostly be accounted for as a weakness of the feature extraction stage. The adaptive Local Subspace Classifier itself was shown to perform better than the equivalent k-Nearest Neighbor classifier.

Even though it seems that the performance of the proposed feature extraction methods and the recognition system as such is not sufficient for the implementation of a PDA system, the present methodology may still be of important advantage. The approach can prove to be beneficial in committee classifiers in which multiple member classifiers are combined to produce the classification decision. In such systems, it is advisable that the errors produced by the members are as independent from each other as possible. Therefore, the combination of a recognizer based on elastic matching and a subsystem based on classification of the pixel images of the characters will be worth an examination.

References

[1] J. Laaksonen. Local subspace classifier. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 637–642, Lausanne, Switzerland, October 1997.

[2] J. Laaksonen. *Subspace Methods in Recognition of Handwritten Digits*. PhD thesis, Helsinki University of Technology, 1997.

[3] J. Laaksonen, V. Vuori, E. Oja, and J. Kangas. Adaptation of prototype sets in on-line recognition of isolated handwritten Latin characters. In S.-W. Lee, editor, *Advances in Handwriting Recognition*. World Scientific Publishing, 1999.

[4] Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, COM-28(1):84–95, January 1980.

[5] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[6] C. C. Tappert. Adaptive on-line handwriting recognition. In *Proceedings of International Conference on Pattern Recognition*, pages 1004–1007. IEEE, 1984.

[7] V. Vuori, J. Laaksonen, E. Oja, and J. Kangas. On-line adaptation in recognition of handwritten alphanumeric characters. In *Proceedings of ICDAR'99*, Bangalore, India, September 1999. To appear.