2009 10th International Conference on Document Analysis and Recognition

Improving a DTW-based Recognition Engine for On-line Handwritten Characters by Using MLPs

M. J. Castro-Bleda S. España J. Gorbe F. Zamora
Depto. de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
46071 Valencia, Spain
{mcastro, sespana, jgorbe, fzamora}@dsic.upv.es

D. Llorens A. Marzal F. Prat J. M. Vilar
Dept. de Llenguatges i Sistemes Informàtics
Universitat Jaume I
12071 Castelló, Spain
{dllorens, amarzal, fprat, jvilar}@lsi.uji.es

Abstract

Our open source real-time recognition engine for on-line isolated handwritten characters is a 3-Nearest Neighbor classifier that uses approximate dynamic time warping comparisons with a set of prototypes filtered by two fast distance-based methods. This engine achieved excellent classification rates on two writer-independent tasks: UJIpenchars and Pendigits. We present the integration of multilayer perceptrons into our engine, an improvement that speeds up the recognition process by taking advantage of the independence of these networks' classification times from training set sizes. We also present experimental results on our new publicly available UJIpenchars2 database and on Pendigits.

1. Introduction

Our real-time recognition engine for isolated handwritten characters is a 3-Nearest Neighbor (3-NN) classifier that uses approximate dynamic time warping (DTW) comparisons with a set of prototypes filtered by two fast distance-based methods. This open source engine, first presented at CCIA 2007 [17], achieved excellent classification rates, presented at VIP 2007 [16], on two writer-independent tasks: UJIpenchars [12, 13] and Pendigits [1]. It is currently employed in our multimodal document processing system STATE [9] for pen input.

We present the integration of multilayer perceptrons (MLPs) into our engine, an improvement that speeds up the recognition process by taking advantage of the independence of these networks' classification times from training set sizes. We also present experimental results on our new publicly available UJIpenchars2 database and on Pendigits, comparing the improved engine with its previous version and with the Microsoft Tablet PC SDK recognizer.

The next section summarizes the previous state of our engine, as presented at VIP 2007. Section 3 discusses the use of MLPs as classifiers and how we integrate them into our engine for speeding up recognition. Our experimental work, showing the effectiveness of such an integration, is presented in Section 4. Finally, some conclusions are drawn in Section 5.

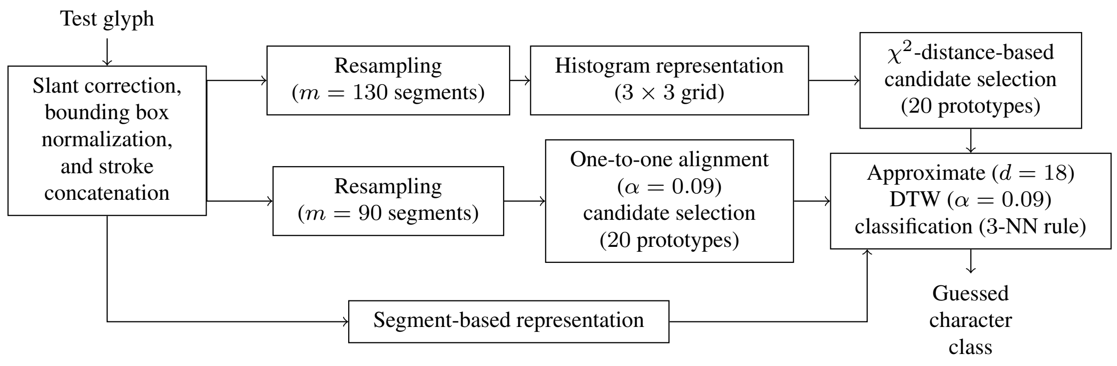
2. Baseline system

As said before, the baseline is our pen-input isolated-character recognition engine as presented at VIP 2007 [16], which will be referred to as the "VIP engine" from now on. The VIP engine uses a 3-NN classifier with an approximate DTW dissimilarity measure, so it needs a corpus of labelled samples, ie prototypes. Since DTW is a rather computationally expensive process, even speeding it up with a Sakoe-Chiba window [19], two different fast screening procedures are applied to the prototypes in a first stage, and only a subset of them is considered by the classifier. The complete recognition process is depicted in Fig. 1.

When a sample has to be classified, it follows a preprocessing procedure consisting of slant correction, a bounding box normalization that preserves the aspect ratio, and stroke concatenation. Then, two different parameterizations are used: a 3×3 histogram of 8-direction codes (72 counts) from a resampling of the joined strokes into 130 segments and a resampling into 90 segments.

The 20 best prototypes according to a χ^2-like distance on the first parameterization are found. Another set of 20 prototypes are selected using a vector distance on the second parameterization. The union of these two sets gives at most 40 prototypes that are compared in turn with the input sample by using a DTW distance, with a Sakoe-Chiba window, on a segment-based representation.

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.209



Test glyph
Slant correction, bounding box normalization, and stroke concatenation
Resampling ($m = 130$ segments)
Histogram representation (3×3 grid)
χ^2-distance-based candidate selection (20 prototypes)
Resampling ($m = 90$ segments)
One-to-one alignment ($\alpha = 0.09$) candidate selection (20 prototypes)
Approximate ($d = 18$) DTW ($\alpha = 0.09$) classification (3-NN rule)
Segment-based representation
Guessed character class

Figure 1. The VIP engine.

Segments are represented by pairs consisting of a point and an angle. Local distance between segments is computed as a linear combination of the squared Euclidean distance between the points and the angular difference affected by a factor $\alpha > 0$.

On the Pendigits [1] task, this engine presents an excellent 0.60% error rate, significantly lower than both the error rate of other recently published techniques [20, 21] and the one we obtain from the Microsoft Tablet PC SDK recognition engine.

This procedure has the handicap that running time depends on the size of the training set since every prototype has to be screened. In the following section, we propose the use of MLPs to alleviate this problem.

3. MLP classifiers and their integration

Artificial neural networks have been widely used in pattern classification and, in particular, in character recognition tasks [3, 5, 8, 10, 11, 14]. The capability of neural networks to estimate the posterior probabilities of the defined classes [4] allows a natural use of these models for classification when labelled data is available.

A particular kind of neural networks, MLPs, have been used in this work. A major requirement to use these models is the fixed dimensionality of the input patterns. For this reason, the original on-line ink information cannot be directly used by the MLPs and an appropriate feature extraction step is needed. We will employ histogram representations since they have a constant number of components and, in [16], provided better results than just resampling. Fig. 2 depicts the computation of one of these histograms for the case of applying a 3×3 grid after a 15-segment resampling.

The architecture of the networks used in this work comprises an input layer with as many neurons as counts in the histogram, one or two hidden layers, and an output layer with one neuron per class. Each layer is fully connected to the next one. Moreover, both hidden and output units have bias. Hidden units are logistic, ie they use the standard sigmoid function, and the output layer uses the softmax activation function [6].

After training, the network can be used in two ways: as a straightforward classifier by selecting the class corresponding to the neuron with the highest score or as a screener by keeping only the c best classes according to their scores. For our purposes, this filtering has an important advantage over prototype-based methods because, once trained, the MLP running time is independent of the training set size.

Summarizing, the idea is to train an MLP that performs a class screening: only prototypes in those classes selected by the MLP will be considered by the VIP engine.

4. Experimental work

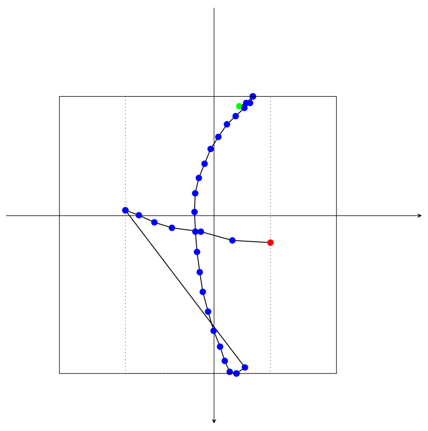
4.1. The databases

We have experimented with two different databases of on-line isolated handwritten characters: UJIpenchars2 and Pendigits.

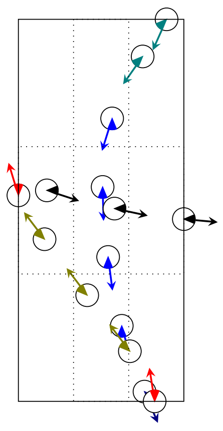
4.1.1. UJIpenchars2. This database [15], available at the UCI Machine Learning Repository [2] and partially described in [13], is an extension of UJIpenchars [12] containing instances of letters, digits, and other symbols.

We have restricted ourselves to the 62 ASCII alphanumeric characters, which have been divided into 35 classes: 9 for the "1" to "9" digits and 26 for the lower and uppercase versions of each letter, where zero is included in the "o" class. In total, we have 4 960 training samples from 40 writers and 2 480 test samples from 20 different writers.

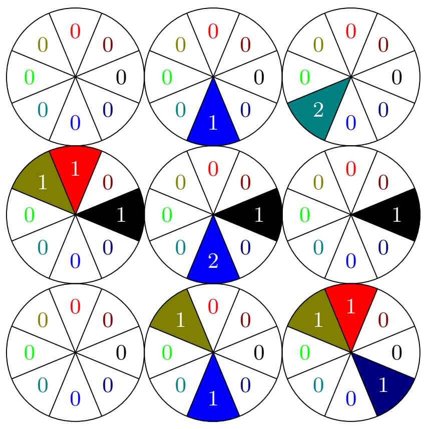
When needed, we have reserved 11 training writers (the UJIpenchars ones) for validation purposes, thus splitting the considered set of samples into three subsets: TR (proper training set, 29 writers, 3 596 samples), VA (validation set, 11 writers, 1 364 samples), and TS (test set, 20 writers, 2 480 samples).



(a) After preprocessing

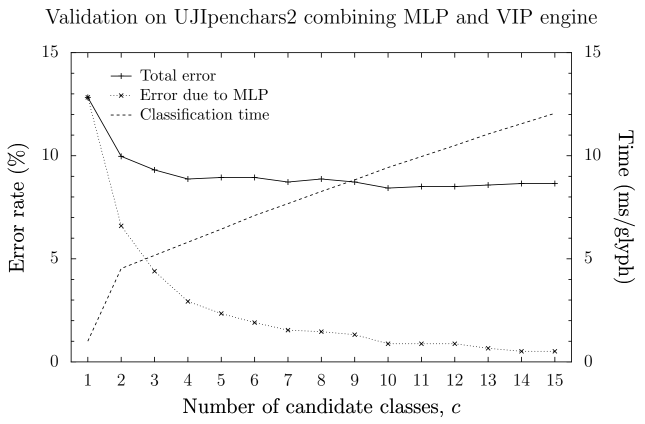


(b) Point and angle pairs



(c) Histogram

Figure 2. Graphical representation of the computation of a histogram for a letter "t".



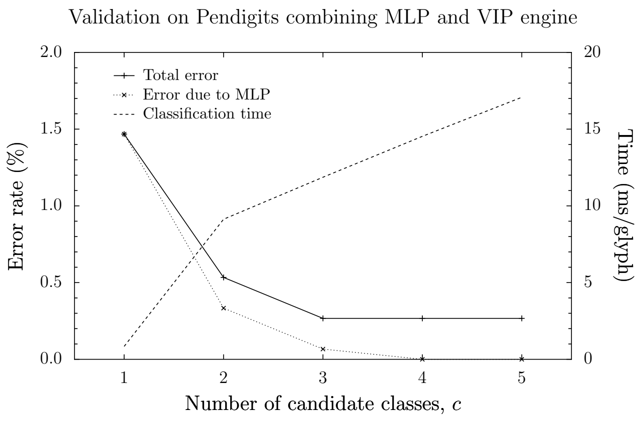


Figure 3. Experimental results for choosing how many candidate classes the MLP must select.

4.1.2. Pendigits. The Pendigits database [1] is also available at the UCI Machine Learning Repository [2]. This corpus contains handwritten instances of the 10 digits from several writers: 7 494 glyphs from 30 writers are used as training set and 3 498 glyphs from 14 different writers are used as test data. Recently published error rates over this database are 2.26% in 2005 [21] and 1.66% in 2006 [20].

When needed, 6 writers have been reserved for validation purposes, producing the following partition: TR (proper training set, 24 writers, 5 995 samples), VA (validation set, 6 writers, 1 499 samples), and TS (test set, 14 writers, 3 498 samples).

4.2. MLP training for UJIpenchars2

For training the MLPs, we have applied the on-line backpropagation algorithm with momentum (see chapters 7 and 8 in [18]) to the cross-entropy penalty function [6] measured on the corresponding TR set. However, to avoid overfitting, the stopping criterion does not take the TR set into account. Instead, the training stops when the classification error on a subset of VA does not improve for 100 epochs. Then, the MLP from the epoch with the minimum classification error so far is chosen.

In this work, all the MLP training has been carried out using the `April` toolkit [7] with scripts that include a small decay to be applied to both learning rate and momentum after each training epoch (they are multiplied by 0.999).

As said before, just a subset of VA is employed for stopping the training of each MLP. This subset, VA1, consists of 774 samples from 6 writers. The remaining 620 validation samples, from 5 writers, forms VA2. The classification error on this second validation set has been used for choosing the best MLP among the networks returned for a number of training runs.

All possible combinations of the following parameter values have been tried in a first round of 2 640 runs (one run per parameter combination):

Resampling sizes: $10, 20, 30, \ldots, 200$.

Grid sizes: 3×3, 4×4, and 5×5.

Hidden layer topologies: One layer with 32, 64, 96, and 128 units and two layers with sizes $(32, 32)$, $(64, 32)$, $(96, 32)$, $(128, 32)$, $(64, 64)$, $(96, 64)$, and $(128, 64)$.

Initial weight ranges: Only $[-0.7, 0.7]$.

Learning rate and momentum pairs: $(0.05, 0.01)$, $(0.01, 0.002)$, $(0.075, 0.025)$, and $(0.005, 0.0001)$.

Then, for the three parameter combinations showing the best results on VA2, a second round of training runs was performed varying the initial weight range. Three intervals ($[-0.1, 0.1]$, $[-0.4, 0.4]$, and $[-0.7, 0.7]$) were considered for each parameter combination and ten different MLPs were trained in each case, totalling 90 new runs. It is worth noting that, in each run, randomness affects both initial weight values and the TR set shuffle before each epoch.

The network finally selected for the UJIpenchars2 task, which achieves a 13.0% error rate on VA2, is an MLP with 128 units in just one hidden layer, trained from weights initialized in the range $[-0.1, 0.1]$ with learning rate 0.05, momentum 0.01, and input patterns obtained from a 4×4 grid after a 70-segment resampling of the preprocessed glyphs.

4.3. MLP training for Pendigits

The same three parameter combinations selected after the first round of runs in the previous section were used for training 90 MLPs for the new task, following the same process described above, except for the fact that there is only one validation set, VA, to be used for both the stopping criterion and choosing the best MLP.

The best performance is achieved by an MLP with two hidden layers $(128, 64)$, learning rate 0.05, momentum 0.01, initial weights in the range $[-0.7, 0.7]$, and input patterns obtained from a 3×3 grid after a 50-segment resampling. The classification error rate for the validation set with this MLP is 1.47%.

4.4. Choosing the number of candidate classes

In order to choose the number c of candidate classes the corresponding MLP should provide to the VIP engine, new experiments were carried out for both UJIpenchars2 and Pendigits. In each case, we use the best MLP previously trained and, as prototypes for the VIP engine, the corresponding TR set. Classification results, measured on the corresponding complete VA set, are shown in Fig. 3 for a range of c values. The times correspond to executions on a Dell Precision T3400 desktop PC with an Intel Core2 Quad CPU Q9450 at 2.66 GHz and 4 Gb of RAM on the .NET 3.5 (with SP1) platform under the Microsoft Vista Ultimate Edition (with SP1) operating system. The programs were coded in C# without unsafe code and compiled on Microsoft Visual Studio 2008 (with SP1).

Our final choice, minimizing classification errors, was $c = 10$ for UJIpenchars2 and $c = 3$ for Pendigits, ie approximately one third of the total number of classes in each case.

4.5. Comparative results

Comparative results are presented in Table 1. The first row corresponds to C# experiments with the 1.7 version of the Microsoft Tablet PC SDK recognition engine[1]. The following rows correspond to the engines explained above, using the same two MLPs as in the previous section. In Validation columns, classification is measured on VA sets and TR sets are used as VIP prototypes. In Test columns, final results on TS sets are shown where, when needed, complete training sets TR + VA are used by the VIP engine.

The main result is that running time has been reduced to 50% from the baseline without significantly altering the error rate: the difference is 0.20% on Pendigits and 0.3% on UJIpenchars2. On UJIpenchars2, the MLP + VIP error rate (8.2%) is clearly competitive with the Microsoft engine (8.5%), while in Pendigits, the error rates of VIP (0.60%) and MLP + VIP (0.80%) are definitely better than those of Microsoft (1.89%).

5. Conclusions

The integration of MLPs into our engine approximately halves its running time while keeping the error rates. Compared to the Microsoft engine, the results are competitive and our code is completely managed, so it can be ported to any architecture running a .NET platform, like Windows, Linux, and Mac OS X, among others.

The current times correspond to more than 60 characters per second, making it suitable for real-time response.

Acknowledgments

Work partially supported by the Spanish *Ministerio de Ciencia e Innovación* (TIN2006-12767, TIN2008-06856, and *Consolider Ingenio 2010* CSD2007-00018), the *Conselleria d'Empresa, Universitat i Ciencia, Generalitat Valenciana* (BFPI06/250 scholarship), and *Fundació Caixa Castelló-Bancaixa* (P1·1B2006-31).

[1]In these experiments, the Microsoft recognition engine uses a `Microsoft.Ink.RecognizerContext` object with an appropriate `WordList` and flags `Coerce` and `WordMode`. We also help the Microsoft recognizer by providing it with the dimensions of the acquisition box via its `Guide` property.

Table 1. Classification error and running time per glyph (ms) on UJIpenchars2 and Pendigits.

	UJIpenchars2				Pendigits			
	Validation		Test		Validation		Test	
Microsoft SDK	8.4%	0.6	8.5%	**0.6**	0.93%	0.5	1.89%	**0.5**
VIP engine	8.7%	16.7	8.5%	23.6	0.27%	26.0	**0.60%**	32.4
MLP	12.8%	1.0	14.2%	1.0	1.47%	0.8	3.63%	0.8
MLP + VIP	8.4%	9.4	**8.2%**	12.4	0.27%	11.9	0.80%	14.8

References

[1] E. Alpaydın and F. Alimoğlu. Pen-Based Recognition of Handwritten Digits (original, unnormalized version). Data set available at [2], 1998.

[2] A. Asuncion and D. Newman. UCI Machine Learning Repository. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2007.

[3] A. Bellili, M. Gilloux, and P. Gallinari. An MLP-SVM combination architecture for offline handwritten digit recognition. *International Journal on Document Analysis and Recognition*, 5(4):244–252, 2003.

[4] C. M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.

[5] M. Blumenstein and B. Verma. A neural based segmentation and recognition technique for handwritten words. In *The 1998 IEEE International Joint Conference on Neural Networks Proceedings*, volume 3, pages 1738–1742, Anchorage, Alaska, USA, 1998.

[6] R. A. Dunne and N. A. Campbell. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In *Proceedings of the Eighth Australasian Conference on Neural Networks*, pages 181–185, Melbourne, Australia, 1997.

[7] S. España-Boquera, F. Zamora-Martínez, M. J. Castro-Bleda, and J. Gorbe-Moya. Efficient BP algorithms for general feedforward neural networks. *Lecture Notes in Computer Science*, pages 327–336, 2007.

[8] P. D. Gader, J. M. Keller, R. Krishnapuram, J.-H. Chiang, and M. A. Mohamed. Neural and fuzzy methods in handwriting recognition. *Computer*, 30(2):79–86, 1997.

[9] A. Gordo, D. Llorens, A. Marzal, F. Prat, and J. M. Vilar. STATE: A multimodal assisted text-transcription system for ancient documents. In *The Eighth IAPR Workshop on Document Analysis Systems*, pages 135–142, Nara, Japan, 2008.

[10] L. Jackel, M. Battista, H. Baird, J. Ben, J. Bromley, C. Burges, E. Cosatto, J. Denker, H. Graf, H. Katseff, Y. LeCun, C. Nohl, E. Sackinger, J. Shamilian, T. Shoemaker, C. Stenard, I. Strom, R. Ting, T. Wood, and C. Zuraw. Neural-net applications in character recognition and document analysis. In *Neural-Net Applications in Telecommunications*. Kluwer Academic Publishers, 1995.

[11] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems 2 (NIPS*89)*, Denver, Colorado, USA, 1990. Morgan Kaufman.

[12] D. Llorens, F. Prat, A. Marzal, and J. M. Vilar. UJIpenchars: A Pen-Based Classification Task for Isolated Handwritten Characters. Data set available as UJI Pen Characters at [2], 2007.

[13] D. Llorens, F. Prat, A. Marzal, J. M. Vilar, M. J. Castro, J. C. Amengual, S. Barrachina, A. Castellanos, S. España, J. A. Gómez, J. Gorbe, A. Gordo, V. Palazón, G. Peris, R. Ramos-Garijo, and F. Zamora. The UJIpenchars Database: A Pen-Based Database of Isolated Handwritten Characters. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, 2008.

[14] I.-S. Oh and C. Y. Suen. A class-modular feedforward neural network for handwriting recognition. *Pattern Recognition*, 35(1):229 – 244, 2002.

[15] F. Prat, M. J. Castro, D. Llorens, A. Marzal, and J. M. Vilar. UJIpenchars2: A Pen-Based Database with More Than 11K Isolated Handwritten Characters. Data set available as UJI Pen Characters (Version 2) at [2], 2008.

[16] F. Prat, A. Marzal, S. Martín, R. Ramos-Garijo, and M. J. Castro. A template-based recognition system for on-line handwritten characters. *Journal of Information Science and Engineering*, 25(3):779–791, 2009.

[17] R. Ramos-Garijo, S. Martín, A. Marzal, F. Prat, J. M. Vilar, and D. Llorens. An input panel and recognition engine for on-line handwritten text recognition. In C. Angulo and L. Godo, editors, *Artificial Intelligence Research and Development*, volume 163 of *Frontiers in Artificial Intelligence and Applications*, pages 223–232. IOS Press, 2007.

[18] R. Rojas. *Neural Networks: A Systematic Introduction*. Springer, 1996.

[19] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

[20] B. Spillmann, M. Neuhaus, H. Bunke, E. Pękalska, and R. P. W. Duin. Transforming strings to vector spaces using prototype selection. *Lecture Notes in Computer Science*, 4106:287–296, 2006.

[21] J. Zhang and S. Z. Li. Adaptive nonlinear auto-associative modeling through manifold learning. *Lecture Notes in Computer Science*, 3518:599–604, 2005.