UDC 004.032.26:004.93

V.V. Romanuke

A 2-LAYER PERCEPTRON PERFORMANCE IMPROVEMENT IN CLASSIFYING 26 TURNED MONOCHROME 60-BY-80-IMAGES VIA TRAINING WITH PIXEL-DISTORTED TURNED IMAGES

There is tried 2-layer perceptron in classifying turn-distorted objects at acceptable classification error percentage. The object model is a letter of English alphabet, which is monochrome 60-by-80-image. Neither training 2-layer perceptron with pixel-distorted images, nor with turn-distorted images makes it classify satisfactorily. Therefore in classifying turn-distorted images a 2-layer perceptron performance might be improved via training under distortion modification. The modified distorted images for the training set are suggested as mixture of turn-distorted and pixel-distorted images. Thus the training set is formed of pixel-distorted turned images on the 26 alphabet letters pattern. A performance improvement is revealed when there are passed much more training samples through 2-layer perceptron. This certainly increases traintime, but instead 2-layer perceptron can classify either of pixel-distorted images and pixel-distorted turned images. At that the trained 2-layer perceptron is about 35 times faster than neocognitron in classifying objects of the considered medium format.

Keywords: automatization, object classification, neocognitron, perceptron, monochrome images, pixel-distortion, rotation, turn-distortion, training set, classification error percentage.

Introduction

Classification is a principal process for automatization. Quality of classification directly influences on how deep automatization may be implemented. Main parameters of the quality are operation speed and accuracy [1, 2].

Neural network may be classifier if it performs, above all, with classification error percentage (CEP) which is not greater than the maximal CEP for an investigated general totality of objects. Operation speed is tolerated secondly. In classifying objects the most accurate neural network is neocognitron [3, 4]. It correctly classifies objects that are differently distorted: shifted, skewed (turned or rotated), scaled. Neocognitron nonetheless is very slow classifier, consuming at that huge informational resources (main memory, hard disk space, high-speed processor). Multilayer perceptron is much faster but ordinarily it classifies objects with shift-skew-scale distortion (SSS-distortion) poorly. Only feature-value-distorted objects are classified perfectly by perceptron if they are not shifted, turned or scaled [1, 5]. For instance, for pixel-distorted images (PDI) the best classifier is 2-layer perceptron (2LP). Moreover, 2LP needs less informational resources.

Thus classification of SSS-distortion objects is still a problem, wherein the bulky neocognitron stands against multilayer perceptron, producing intolerably high CEP. Hereby, if 2LP could perform on SSS-distortion objects with acceptable CEP for some classification problems, it would be admissible to substitute neocognitron with 2LP in these problems, and this would ensure the high productivity up with low resources consumption and classifier short response delay. At least one of distortion types from SSS-distortion might be associated with objects, which are to be classified by 2LP or perceptron with more layers. For that try, further, let there be the image as an object model. In particular, an image, feeding the classifier input, may be slightly rotated or turned through an angle as it occurs usually while scanning or photographing and retrieving information.

Perfectiveness of 2LP in classifying PDI might be adjusted for classifying turn-distorted images (TDI), like never before [1, 2, 5]. Such adjustment could be realized via training 2LP with pixel-distorted turned images (PDTI). The perceptron can be constructed and trained within MATLAB environment, using accustomed MATLAB scripts and functions from MATLAB Neural Network Toolbox. This environment will be used also for testing the trained 2LP to determine its functionality in classifying TDI, where bulky neocognitron takes from several seconds up to a minute to classify an object of TDI type of medium format [6].

The simplest image model is monochrome image, modeled as matrix of logical elements. Subsequently, there should be patterned the general totality of monochrome images, and representatives of all the classes, whereupon 2LP is going to be trained with PDI, TDI, PDTI, and tested on as TDI, as well as PDTI. Then there can be revealed the trained 2LP producing the lowest CEP, whether it is acceptable or not.

Problem statement

The aim of this paper is to reach a 2-layer perceptron performance improvement in classifying turned monochrome 60-by-80-images via training with pixel-distorted turned images on 26 alphabet letters pattern.

A pattern for general totality and the representatives

It is obvious that for obtaining adequate investigation results, the monochrome image shouldn't be of small format. On the other side, they shouldn't be large-format for accelerating investigation procedures. Regarding this, an appropriate format for the monochrome image is 60×80. So, now there is general totality of 60×80 matrices of ones and zeros.

Further, a satisfactory fine monochrome image pattern is the letter, capitalized or not, no matter. Henceforward, the representatives of 26 classes are matrices $\{\mathbf{A}_q\}_{q=1}^{26}$, where q-th image as the q-th class representative is modeled as matrix $\mathbf{A}_q=(a_{uv}^{\langle q\rangle})_{60\times80}$ of $a_{uv}^{\langle q\rangle}\in\{0,1\}$ by $q=\overline{1,26}$. The representative is a bitmap file, viewed easily from within MATLAB (figure 1), where, appositely, the white color is coded with ones.

Now PDI, TDI, PDTI are to be modeled for MATLAB environment, and then 2LP will be trained with training samples, formed on the basis of these models. It's a next step in revealing the possibility of the perceptron to classify those image objects that might have been seen that only neocognitron can.

Training by model of PDI

Inasmuch as 2LP is trained with blocks of feature-vectorized objects, then the q-th class representative as matrix $\mathbf{A}_q=(a_{uv}^{\langle q\rangle})_{60\times80}$ is reshaped into 4800-length-column, whereupon 26 columns are concatenated horizontally into the matrix $\mathbf{A}=(\overline{a}_{jq})_{4800\times26}$ of all representative 26 monochrome 60×80 images. The q-th column of the matrix $\mathbf{A}=(\overline{a}_{jq})_{4800\times26}$ is the column-reshaped representative of the q-th class. Then model of PDI is just the matrix

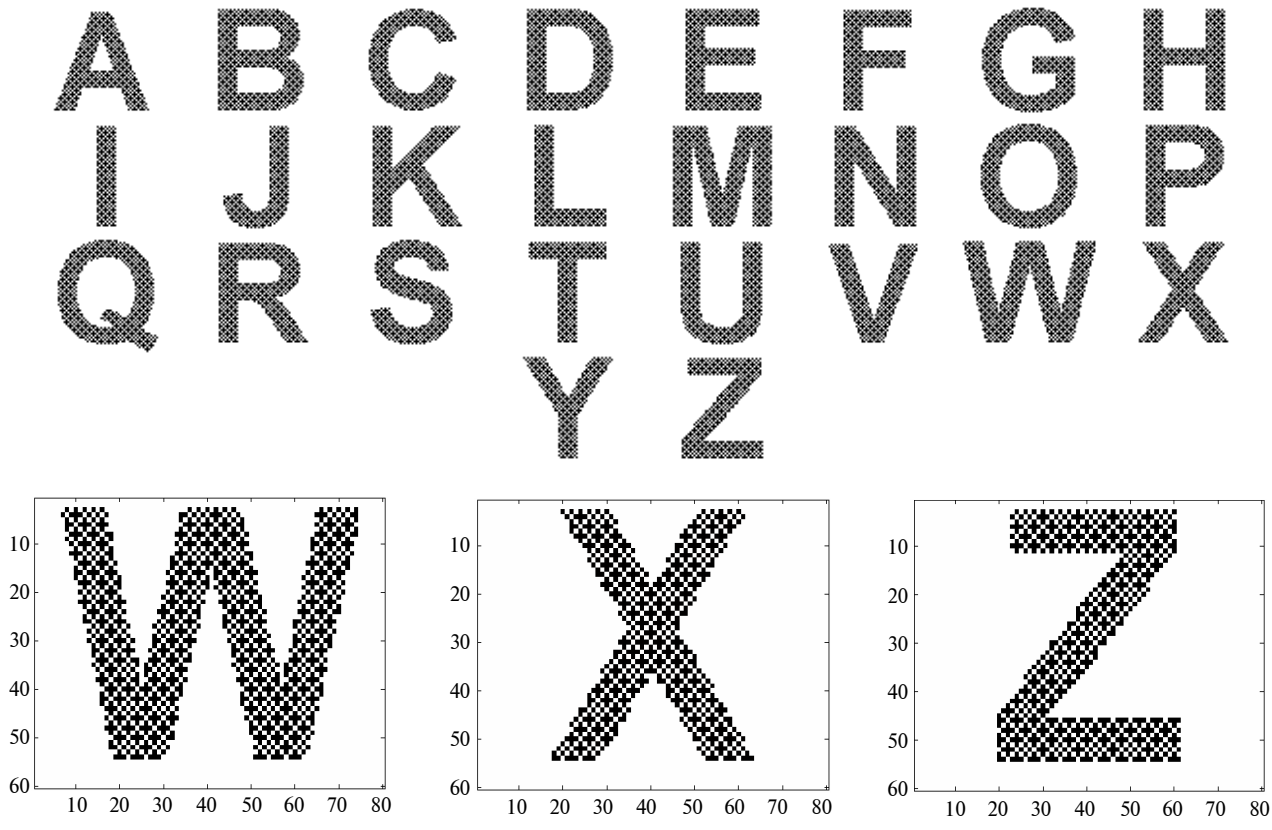

Figure 1. Monochrome 60×80 images of 26 English alphabet letters, viewed as bitmap files, and letters "W", "X", "Z", viewed within MATLAB (it can be seen that the letter black cast is sprinkled crosshatch-regularly with white specks)

$$\mathbf{A}_{\text{pixel}}^{\langle k\rangle} = \mathbf{A} + \sigma_{\text{pixel}}^{\langle k\rangle} \cdot \Xi \qquad (1)$$

by standard deviation (SD)

$$\sigma_{\text{pixel}}^{\langle k\rangle} = \sigma_{\text{pixel}}^{\langle \max\rangle} F^{-1} k \quad \forall k = \overline{1, F} \qquad (2)$$

and its maximum $\sigma_{\text{pixel}}^{\langle \max\rangle} > 0$ at 4800×26-matrix Ξ of values of normal variate with zero expectation and unit variance (ZEUV). In the assignment (2) the number F indicates at smoothness in training the perceptron [7].

For the training process, the input of 2LP is fed with the training set

$$\tilde{P}_{\text{train}} = \{\tilde{\mathbf{P}}_i\}_{i=1}^{C+F} = \{\{\mathbf{A}\}_{l=1}^{C}, \{\mathbf{A}_{\text{pixel}}^{\langle k\rangle}\}_{k=1}^{F}\} \qquad (3)$$

of C replicas of all classes representatives and PDI (pixel-distorted representatives) by the set of identifiers (targets)

$$T = \{\mathbf{T}_i\}_{i=1}^{C+F} = \{\mathbf{I}\}_{i=1}^{C+F} \qquad (4)$$

with identity 26×26-matrix $\mathbf{I}$. The set (3), being formed by (1) and (2), is passed through 2LP with targets (4) for Q_{pass} times. For accomplishing the training process there is available MATLAB training function “traingda”, being one of the fastest for 2LP, trained with backpropagation algorithm [1, 8, 9].

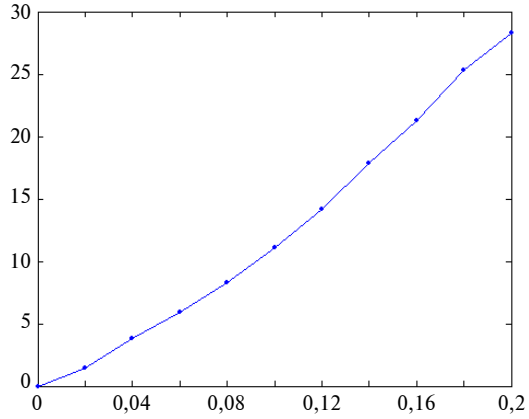

Figure 2. Near-linearly increasing percentage p_{error} of classification errors over SD range $[0; \sigma_{\text{turn}}^{\langle \max\rangle}]$ of the turn angle intensity in TDI by 250 neurons in 2LP hidden layer and $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, derived from 2000 batch testings of 2LP trained with PDI at $\sigma_{\text{pixel}}^{\langle \max\rangle} = 1$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 10$ (traintime duration is about 618 seconds)

Having set the size of 2LP hidden layer to 250 neurons at $\sigma_{\text{pixel}}^{\langle \max\rangle} = 1$, $C = 2$, $F = 8$ for (1)–(4), the results of classifying TDI by 2LP trained with PDI in (3), passed for $Q_{\text{pass}} = 10$ times, appear quite unacceptable (figure 2). These results are obtained in routine of the batch testing of 2LP. And the results of the letter-by-letter testing of 2LP at the highest SD for (1) disclose that distribution of CEP over letters is far from uniform (figure 3).

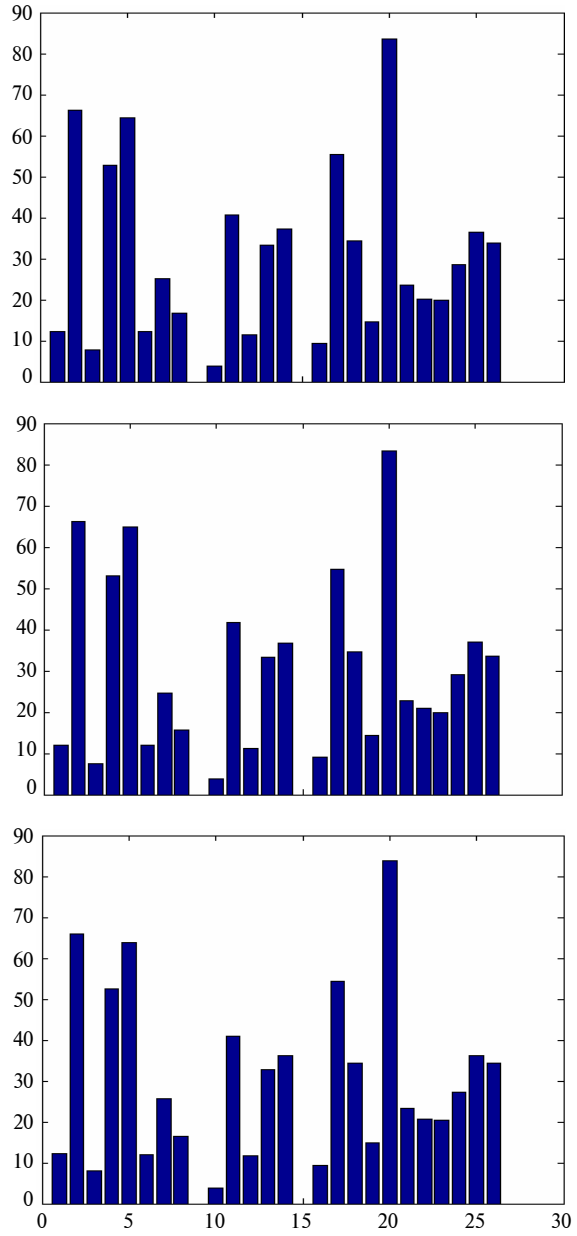

Figure 3. Distributions of CEP over letters at fixed SD $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, derived from 10000 testings of 2LP trained with PDI at $\sigma_{\text{pixel}}^{\langle \max\rangle} = 1$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 10$

Consequently, 2LP trained by model of PDI, cannot classify TDI. Actually, there is no something unexpected, because perceptron paradigm is

that it can classify only that, what feeds its input while training. And it's likely that the next section will allow to clear up the question of classifying TDI, when 2LP is trained by model of TDI.

Training by model of TDI

Model of TDI cannot be applied to the batch of images, like it is in the addition (1). Here, having assumed some SD of turn-distortion, every image is turned through the angle, determined separately under this deviation, multiplied by a value of ZEUV normal variate. Such model, realized within MATLAB as a function "imrotate", for rotating the image $\mathbf{A}_q$ is stated implicitly as

$$\tilde{\mathbf{A}}_q(k) = 1 - \tau(1 - \mathbf{A}_q,\ \beta(\sigma_{\text{turn}}^{\langle k\rangle}),\ M,\ S). \qquad (5)$$

The map τ in (5) along with $\mathbf{A}_q$ takes the following arguments: the turn angle $\beta(\sigma_{\text{turn}}^{\langle k\rangle})$ by SD

$$\sigma_{\text{turn}}^{\langle k\rangle} = \sigma_{\text{turn}}^{\langle \max\rangle} F^{-1} k \quad \forall\ k = \overline{1, F} \qquad (6)$$

for $\sigma_{\text{turn}}^{\langle \max\rangle} > 0$ at k-th part of forming the set that will feed the input of 2LP, and handles M and S. The map (5) rotates the input image $1 - \mathbf{A}_q$ by

$$\beta(\sigma_{\text{turn}}^{\langle k\rangle}) = \frac{180}{\pi} \sigma_{\text{turn}}^{\langle k\rangle} \xi(k) \qquad (7)$$

degrees around its center point, where $\xi(k)$ is value of ZEUV normal variate, raffled at the k-th stage of TDI set formation. The image is turned in counterclockwise direction if $\beta(\sigma_{\text{turn}}^{\langle k\rangle}) > 0$. For $\beta(\sigma_{\text{turn}}^{\langle k\rangle}) < 0$ the image is turned clockwise; for $\beta(\sigma_{\text{turn}}^{\langle k\rangle}) = 0$ the image remains unturned. It is important, MATLAB function "imrotate" makes the output image large enough to contain the entire rotated image, where nearest neighbor interpolation is used by default, setting the values of pixels in the output image that are outside the rotated image to zero. In MATLAB, zero value is imaged as the black, so that is why the rotation is applied to the inverted image $\mathbf{A}_q$, and the output image of the map τ in (5) is inverted back. The value M is the handle to the interpolation method, and the value S is the handle for specifying the size of the returned image. Within MATLAB function "imrotate" the interpolation method is set by a text string that can have one of the following values: "nearest" (nearest-neighbor interpolation), "bilinear" (bilinear interpolation), "bicubic" (bicubic interpolation, what can produce pixel values outside the original range). Also MATLAB function "imrotate" allows to set the size of the returned image with a text string that can have either of the following values: "crop" (makes output image be the same size as the input image, cropping the rotated image to fit), "loose" (default value, where it makes output image be large enough to contain the entire rotated image).

After all 26 images have become TDI, each q-th image as matrix $\tilde{\mathbf{A}}_q(k) = [\tilde{a}_{uv}^{\langle q\rangle}(k)]_{60\times 80}$ is reshaped into 4800-length-column, $q = \overline{1, 26}$. Then the matrix $\mathbf{A}_{\text{turn}}^{\langle k\rangle} = [\overline{\tilde{a}}_{jq}(k)]_{4800\times 26}$ of all 26 TDI, reshaped into 26 columns, is included into the training set

$$\tilde{P}_{\text{train}} = \{\tilde{\mathbf{P}}_i\}_{i=1}^{C+F} = \{\{\mathbf{A}\}_{l=1}^{C}, \{\mathbf{A}_{\text{turn}}^{\langle k\rangle}\}_{k=1}^{F}\} \qquad (8)$$

that feeds the input of 2LP, passing through 2LP with targets (4) for Q_{pass} times.

Once again, for setting the size of hidden layer of 2LP to 250 neurons at $\sigma_{\text{turn}}^{\langle \max\rangle} = 0{,}2$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 10$, and using the training MATLAB-function "traingda", the results of classifying TDI, when 2LP is trained with TDI and batch-tested, appear unsatisfactory (figure 4) to higher SD. However, now these results are much better than those ones, derived from 1000 batch testings of PDI-trained 2LP in figure 2 and figure 3, although the results of the letter-by-letter testing

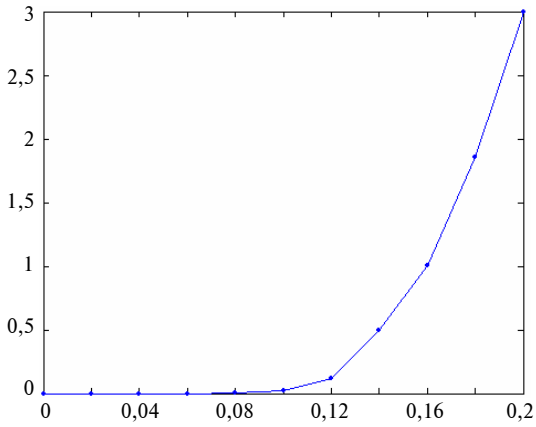

Figure 4. Percentage p_{error} of classification errors over SD range $[0; \sigma_{\text{turn}}^{\langle \max\rangle}]$ by 250 neurons in 2LP hidden layer and $\sigma_{\text{turn}}^{\langle \max\rangle} = 0{,}2$, derived from 2000 batch testings of 2LP trained with TDI at $\sigma_{\text{turn}}^{\langle \max\rangle} = 0{,}2$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 10$ (traintime duration is about 1239 seconds)

of TDI-trained 2LP at the highest SD for (7) still are non-uniform (figure 5). But the training process for TDI is running about at least twice slower than for PDI, and the averaged CEP, being much lower than for PDI-trained 2LP, nonetheless remains high at the maximal SD. Undesirably, it abruptly increases just after SD is 0,1.

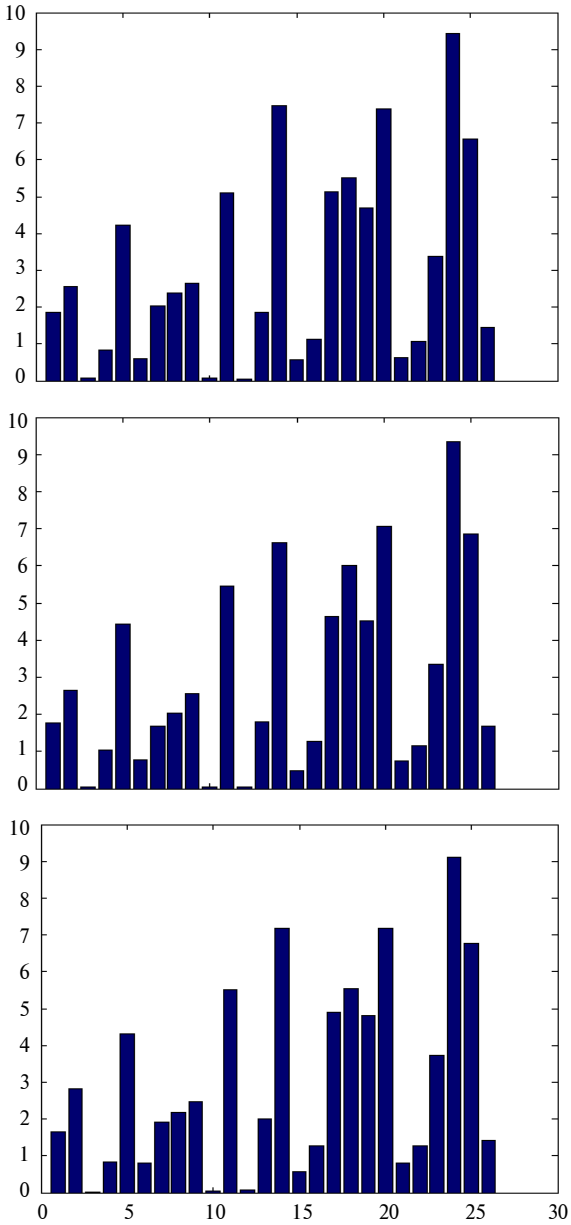

Figure 5. Distributions of CEP over letters at fixed SD $\sigma_{\text{turn}}^{\langle\max\rangle} = 0,2$, derived from 10000 testings of 2LP trained with TDI at $\sigma_{\text{turn}}^{\langle\max\rangle} = 0,2$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 10$

As it is seen the performance of 2LP trained by model of TDI is that such 2LP can classify TDI satisfactorily only at insignificant SD. Acceleration of its training process, being some lingered, is unlikely. Besides, 2LP trained with TDI cannot classify PDTI (figure 6), formed as

$$\mathbf{A}_{\text{pixel-turn}}^{\langle k\rangle} = \mathbf{A}_{\text{turn}}^{\langle k\rangle} + \sigma_{\text{pixel}}^{\langle k\rangle} \cdot \Xi \qquad (9)$$

at some k, meaning its non-universality. By the way, PDTI occur more often than TDI. Consequently, there is a need of modifying the type of distortion in training 2LP for to make it classify as TDI, as well as PDTI.

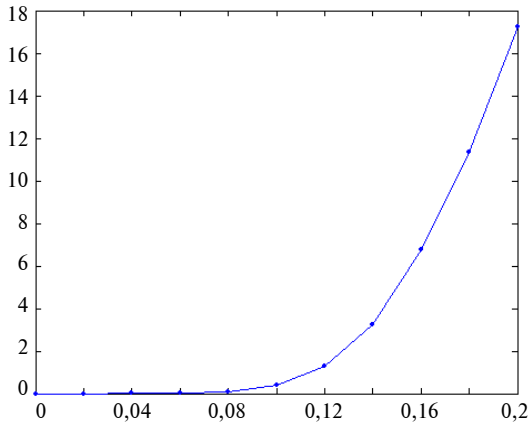

Figure 6. Percentage p_{error} of classification errors over SD range $[0; \sigma_{\text{turn}}^{\langle\max\rangle}]$ by 250 neurons in 2LP hidden layer and $\sigma_{\text{turn}}^{\langle\max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle\max\rangle} = 5\sigma_{\text{turn}}^{\langle\max\rangle}$, $\sigma_{\text{turn}}^{\langle k\rangle} \in [0; \sigma_{\text{turn}}^{\langle\max\rangle}]$, $\sigma_{\text{pixel}}^{\langle k\rangle} \in [0; \sigma_{\text{pixel}}^{\langle\max\rangle}]$, derived from 2000 batch testings of 2LP trained with TDI at $\sigma_{\text{turn}}^{\langle\max\rangle} = 0,2$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 10$

The idea of the distortion modification is in making the training set with mixture of TDI and PDI. There is a hope that such intermediary training set between turn-distortion and pixel-distortion will acquire as feature of faster training process for PDI, as well as feature of classifying either TDI or PDTI. However, this probably may take greater number of passes Q_{pass}.

Training by model of PDTI

In model of PDTI the input of 2LP is fed with the training set

$$\tilde{P}_{\text{train}} = \{\tilde{\mathbf{P}}_i\}_{i=1}^{C+F} = \{\{\mathbf{A}\}_{l=1}^{C}, \{\mathbf{A}_{\text{pixel-turn}}^{\langle k\rangle}\}_{k=1}^{F}\} \qquad (10)$$

of C replicas of undistorted images and PDTI in matrices (9) by the set of targets (4). Once again, the set (10) is passed through 2LP for Q_{pass} times,

and this number now should be much greater than if it were model of TDI or PDI.

At k-th part of forming the set (10) let $\sigma_{\text{pixel}}^{\langle k\rangle} = 5\sigma_{\text{turn}}^{\langle k\rangle}$ by $k = \overline{1,F}$ and $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle \max\rangle} = 5\sigma_{\text{turn}}^{\langle \max\rangle}$. After having been trained with the set (10) for $Q_{\text{pass}} = 200$, 2LP produce yet higher performance than 2LP trained with TDI (figure 7) under same parameters. Meanwhile, herein for $Q_{\text{pass}} = 10$ the traintime duration is about 1,25 times shorter (996 seconds) than for training with TDI (figure 4), what means that 2LP is trained faster over PDTI, and it persuades in reasonableness of making Q_{pass} as great as needed. Training 2LP for $Q_{\text{pass}} = 300$ and $Q_{\text{pass}} = 400$ makes it be expectedly more effective (figure 8). Testing of PDTI-trained 2LP at the highest SD for

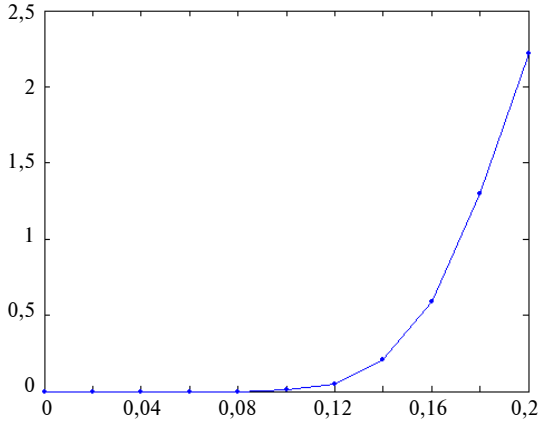

Figure 7. Percentage p_{error} of classification errors over SD range $[0; \sigma_{\text{turn}}^{\langle \max\rangle}]$ by 250 neurons in 2LP hidden layer and $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, derived from 2000 batch testings of 2LP trained with PDTI at $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle \max\rangle} = 5\sigma_{\text{turn}}^{\langle \max\rangle}$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 200$ (traintime duration is about 4588 seconds)

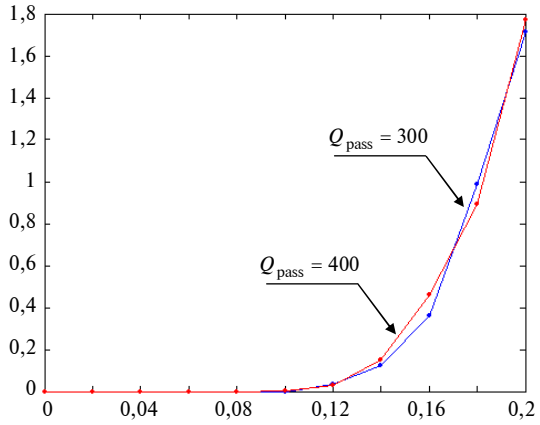

Figure 8. CEP p_{error} over SD range $[0; \sigma_{\text{turn}}^{\langle \max\rangle}]$ by 250 neurons in 2LP hidden layer and $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, derived from 2000 batch testings of 2LP trained with PDTI at $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle \max\rangle} = 5\sigma_{\text{turn}}^{\langle \max\rangle}$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 300$ (traintime duration is about 7051 seconds) and $Q_{\text{pass}} = 400$ (traintime duration is about 8053 seconds)

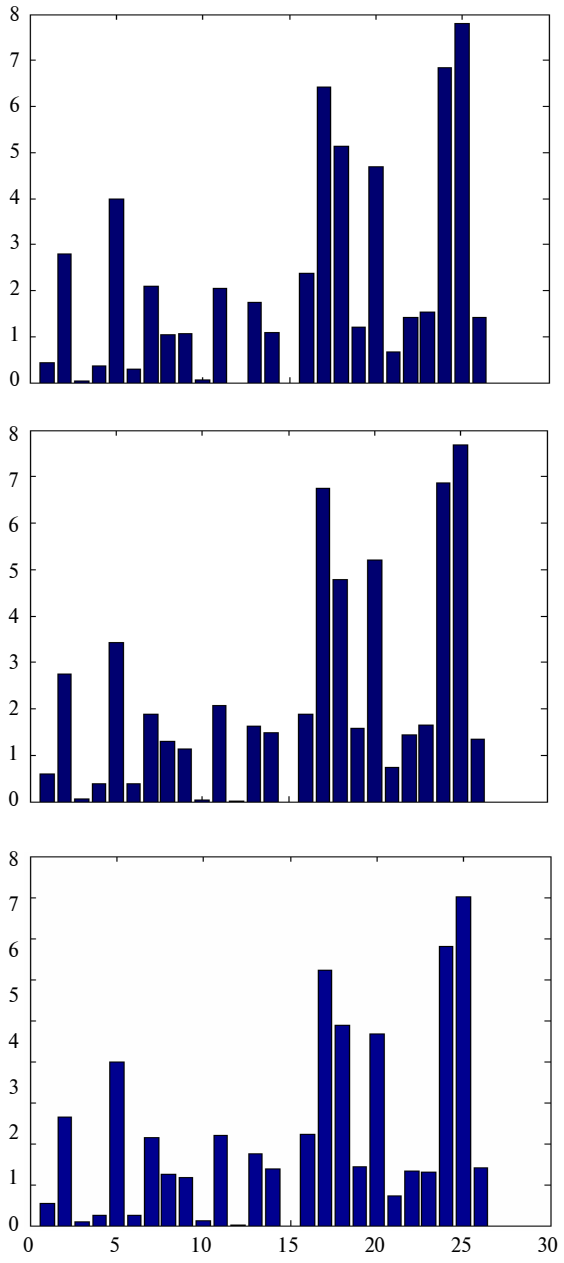

Figure 9. Distributions of CEP over letters at fixed SD $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, derived from 10000 testings of 2LP trained with PDTI at $\sigma_{\text{turn}}^{\langle \max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle \max\rangle} = 5\sigma_{\text{turn}}^{\langle \max\rangle}$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 200$

classifying TDI letter-by-letter certifies it (figure 9), although non-uniformity of distributions of CEP doesn't vanish. Another gain of training 2LP with PDTI is that now 2LP classifies PDTI much better (figure 10). Nevertheless, after SD is 0.1 the percentage p_{error} of classification errors is anyway increasing steeply.

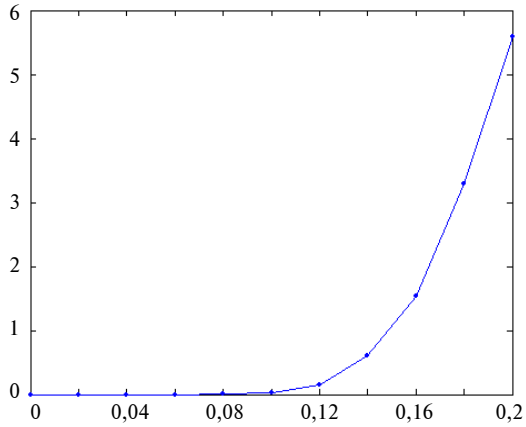

Figure 10. Percentage p_{error} of classification errors over SD range $[0;\ \sigma_{\text{turn}}^{\langle\max\rangle}]$ by 250 neurons in 2LP hidden layer and $\sigma_{\text{turn}}^{\langle\max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle\max\rangle} = 5\sigma_{\text{turn}}^{\langle\max\rangle}$, $\sigma_{\text{turn}}^{\langle k\rangle} \in [0;\ \sigma_{\text{turn}}^{\langle\max\rangle}]$, $\sigma_{\text{pixel}}^{\langle k\rangle} \in [0;\ \sigma_{\text{pixel}}^{\langle\max\rangle}]$, derived from 2000 batch testings of 2LP trained with PDTI at $\sigma_{\text{turn}}^{\langle\max\rangle} = 0,2$, $\sigma_{\text{pixel}}^{\langle\max\rangle} = 5\sigma_{\text{turn}}^{\langle\max\rangle}$, $C = 2$, $F = 8$, $Q_{\text{pass}} = 200$

The classifier response delay is all right: 2LP (PDTI-trained or other) takes about 43 seconds to classify 1000 letters on CPU, and about 25 seconds on GPU [10, 11] at the corresponding CEP (RAM is not less than one gigabyte). Values p_{error} can be lowered further just if Q_{pass} is greater and the training process with PDTI is prolonged. If the traintime duration is not critical then seemingly that CEP of 2LP trained with PDTI can be made as low as needed.

Conclusion

Having patterned the general totality of monochrome 60×80 images as the classifier input, and taken 26 monochrome 60×80 images of English alphabet letters as representatives of all the classes, there is a method of lowering the percentage p_{error} of classification errors when TDI or PDTI are on the input of 2LP. The method core is in training 2LP with the training set, accumulated both TDI and PDI. Such training set (10) with TDI by its model (5)–(7) for reshaping matrices $\{\tilde{\mathbf{A}}_q(k)\}_{q=1}^{26}$ into $\mathbf{A}_{\text{turn}}^{\langle k\rangle} = [\bar{\tilde{a}}_{jq}(k)]_{4800\times26}$ and their subsequent usage in PDTI model (9) has been accumulated at SD ratio $\sigma_{\text{pixel}}^{\langle\max\rangle} = 5\sigma_{\text{turn}}^{\langle\max\rangle}$. This ratio is heuristic and probably nonoptimal, but it's giving the acceptable CEP in classifying either TDI or PDTI. Perhaps, the lowest CEP may be reached at when the ratio is optimized. As to results of the letter-by-letter testing of 2LP then nonuniformity in distribution of CEP over letters cannot be smoothed with ordinary methods. The important finding is that 2LP can classify TDI, though needing much greater times of passing the training set through itself, but being far beyond faster (it is about 35 times faster than neocognitron in classifying objects of the considered medium format) and lighter than neocognitron. And inasmuch as here the letter monochrome image has been a model of the object, these findings easily extend over general totalities of many other images (even colored) of any format for any number of classes. A promising outlook for further investigation gravitates toward attaching TDI to other classes of distorted images and trying to associate them with the pixel distortion, what would let project smarter 2LP classifiers of SSS-distortion objects.

References

1. *S. Haykin*, Neural Networks: A Comprehensive Foundation. New Jersey: Prentice Hall, Inc, 1999.
2. *G. Arulampalam and A. Bouzerdoum*, "A generalized feedforward neural network architecture for classification and regression", Neural Networks, vol. 16, no. 5-6, pp. 561–568, 2003.
3. *K. Fukushima*, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", Biological Cybernetics, vol. 36, no. 4, pp. 193–202, 1980.
4. *K. Fukushima*, "Neocognitron: A hierarchical neural network capable of visual pattern recognition", Neural Networks, vol. 1, no. 2, pp. 119–130, 1988.
5. *K. Hagiwara et al.,* "Upper bound of the expected training error of neural network regression for a Gaussian noise sequence", Ibid, vol. 14, no. 10, pp. 1419–1429, 2001.

6. *G. Poli and J.H. Saito*, Parallel Face Recognition Processing using Neocognitron Neural Network and GPU with CUDA High Performance Architecture, in Face Recognition, M. Oravec, Ed., InTech, 2010.
7. *Романюк В.В.* Зависимость производительности нейросети с прямой связью с одним скрытым слоем нейронов от гладкости ее обучения на зашумленных копиях алфавита образов // Вісник Хмельницького нац. ун-ту. Технічні науки. – 2013. – № 1. – С. 201–206.
8. *M.T. Hagan and M.B. Menhaj,* "Training feedforward networks with the Marquardt algorithm", IEEE Trans. Neural Networks, vol. 5, no. 6, pp. 989–993, 1994.
9. *A. Nied et al.,* "On-line neural training algorithm with sliding mode control and adaptive learning rate", Neurocomputing, vol. 70, no. 16-18, pp. 2687–2691, 2007.
10. *K.-S. Oh and K. Jung*, "GPU implementation of neural networks", Pattern Recognition, vol. 37, no. 6, pp. 1311–1314, 2004.
11. *D. Kangin et al.,* "Further Parameters Estimation of Neocognitron Neural Network Modification with FFT Convolution", J. Telecomm., Electronic and Comp. Eng., vol. 4, no. 2, pp. 21–26, 2012.

Рекомендована Радою
факультету прикладної математики
НТУУ "КПІ"

Надійшла до редакції
27 січня 2014 року