DE GRUYTER OPEN 1862

Applied Computer Systems doi: 10.1515/acss-2016-0008

2016/19

Optimal Pixel-to-Shift Standard Deviation Ratio for Training 2-Layer Perceptron on Shifted 60×80 Images with Pixel Distortion in Classifying Shifting-Distorted Objects

Vadim V. Romanuke

Khmelnitskiy National University, Ukraine

***Abstract* – An optimization problem of classifying shifting-distorted objects is studied. The classifier is 2-layer perceptron, and the object model is monochrome 60×80 image. Based on the fact that previously the perceptron has successfully been attempted to classify shifted objects with a pixel-to-shift standard deviation ratio for training, the ratio is optimized. The optimization criterion is minimization of classification error percentage. A classifier trained under the found optimal ratio is optimized additionally. Then it effectively classifies shifting-distorted images, erring only in one case from eight takings at the maximal shift distortion. On average, classification error percentage appears less than 2.5 %. Thus, the optimized 2-layer perceptron outruns much slower neocognitron. And the found optimal ratio shall be nearly the same for other object classification problems, when the number of object features varies about 4800, and the number of classes is between two and three tens.**

***Keywords* – 2-layer perceptron, classification error percentage minimisation, monochrome images, object classification, optimal training, shifting-distorted objects.**

I. Problem of Classifying Shifting-Distorted Objects

When monitoring and controlling processes are running, automatization system clashes with distortions in the objects being monitored. A very corrupting distortion type ties in shift distortion. A shifting-distorted object (SDO) is hard to recognise immediately. This is explained by the fact that shift distortion is a specific feature distortion type, which cannot be modelled simply as a recognition noise for applying rapid perceptrons subsequently. Much slower neocognitrons classify SDO instead [1], [2]. An attempt for 2-layer perceptron (2LP) high performance in classifying SDO via training it with normally noised objects (NNO) on the pattern of 26 alphabet letters was made in [3]. As it was authorised, if the training process of 2LP were configured optimally then 2LP would be capable of classifying SDO nearly at classification error percentage (CEP), which is performed by neocognitrons [4], [5]. With 2LP for classifying SDO, substantial loss is just the training process duration (TPD).

II. Training the Perceptron on Shifted Objects

If 2LP is trained purely on shifted objects, without any additional supplementations into training sets to feed the input of 2LP, TPD may drag on endlessly [3], [6], [7]. To shorten TPD, there are sets of NNO that ought to supplement the training set, feeding the input of 2LP [8], [9]. In [3], with the pattern of 26 enlarged English alphabet capital letters (EEACL) the object was modelled [10], [11] as monochrome 60×80 image (M6080I). Shifting-distorted M6080I (SDM6080I) was defined with shifting standard deviation (SSD)

$$\sigma_{\text{SDM6080I}}^{\langle k\rangle}=\frac{\sigma_{\text{SDM6080I}}^{\langle\max\rangle}}{F}\cdot k \quad \text{at} \quad k=\overline{1,F} \tag{1}$$

on k-th epoch of forming the set of SDM6080I (k-EFSDM6080I) by the number $F\in\mathbb{N}$ (smoothness in training [9]) and a maximal SSD $\sigma_{\text{SDM6080I}}^{\langle\max\rangle}$. SSD (1) on k-EFSDM6080I along with raffling the normal variate with zero expectation and unit variance (NVZEUV) defined how many pixels an M6080I was shifted horizontally and vertically. For SDM6080I, the method of making NNO was forming SDM6080I with pixel distortion (SDM6080IPD). SDM6080IPD was defined with pixel distortion SSD (PDSSD)

$$\sigma_{\text{SDM6080IPD}}^{\langle k\rangle}=\frac{\sigma_{\text{SDM6080IPD}}^{\langle\max\rangle}}{F}\cdot k \quad \text{at} \quad k=\overline{1,F} \tag{2}$$

on k-th epoch of forming the set of SDM6080IPD (k-EFSDM6080IPD) by a maximal PDSSD $\sigma_{\text{SDM6080IPD}}^{\langle\max\rangle}$. PDSSD (2) on k-EFSDM6080IPD along with NVZEUV defined how each of 4800 pixels of an SDM6080I was changed from its initial state value (0 or 1) to a real value (probably, excluding irrationals). As seen from (1) and (2), 2LP was trained with SDM6080IPD under some ratio of maximal PDSSD and SSD

$$r=\frac{\sigma_{\text{SDM6080IPD}}^{\langle\max\rangle}}{\sigma_{\text{SDM6080I}}^{\langle\max\rangle}}. \tag{3}$$

Pixel-to-shift standard deviation ratio (PSSDR) in (3) was the constant for the given 2LP and SDO classification problem. It was stated that the better PSSDR (3) was adjusted [12], [13] the lower CEP was going to be performed by 2LP in classifying SDO, modelled here as SDM6080I. Under nonoptimal PSSDR, training the perceptron on SDO was like TPD blind dragging that could continue and oscillate for nought, just like 2LP was unsuccessfully trained purely on shifted objects.

III. The Paper Aim and Task Formulation

Suppose that $v_{\text{CEP}}(r)$ is a value of CEP in classifying SDM6080I performed by 2LP that was trained with SDM6080IPD under PSSDR (3). Namely, CEP is the multiplied-by-100 ratio of the amount of misclassified objects to the total object amount fed the 2LP input. And $r_{\min}$ is minimally tolerable PSSDR, $r_{\max}$ is maximally tolerable PSSDR for this classification problem. If $\forall\, r \in [r_{\min}; r_{\max}]$ every value $v_{\text{CEP}}(r)$ is known then the aim is to solve the problem

$$r^* \in \arg \min_{r \in [r_{\min};\, r_{\max}]} v_{\text{CEP}}(r). \tag{4}$$

Stochastic function $v_{\text{CEP}}(r)$ having its unknown continuous average on the segment $[r_{\min}; r_{\max}]$ is to be evaluated rather than approximated. The tasks to be accomplished are as follows:

1. To define general totality (GT) and non-distorted representatives (NDR) of its classes. The c-th NDR is supposed to be the c-th EEACL in the list of alphabetically ordered M6080I of those 26 EEACL, $c = \overline{1, 26}$.

2. To preset the number of neurons in perceptron hidden layers and a method for training it. Since 2LP has the single hidden layer (SHL), the number of neurons N_{SHL} in SHL must be appointed. For modelling and simulating neural networks, MATLAB environment is the most applicable one. Then a MATLAB function for training perceptrons should be selected.

3. To put statements for a model of SDM6080IPD.

4. To determine the range $[r_{\min}; r_{\max}]$ of PSSDR.

5. To run through the range of PSSDR in order to evaluate the function $v_{\text{CEP}}(r)$.

6. To minimise the averaged CEP for 2LP due to the problem (4), thus allowing effectively classifying SDO on the pattern of the defined GT.

IV. GT and NDR of Its Classes

If there were no M6080I with pixel distortion, then there would be objects in GT whose model would be 60×80 matrix of zeros and ones (MZO). Such GT would be finite and have 2^{4800} elements, among which there would be 26 MZO, imaging 26 EEACL as NDR. Other $2^{4800} - 26$ MZO would be either SDM6080I of EEACL or M6080I that would not be EEACL. Real GT, containing SDM6080IPD with 26 NDR $\left\{ \mathbf{X}_c = \left(x_{uv}^{\langle c \rangle} \right)_{60 \times 80} \right\}_{c=1}^{26}$ and SDM6080I, is infinite: 2^{4800} MZO are supplemented with continuum of 60×80 matrices of real values without excluded irrationals (MRVEI). Henceforward, GT with 26 NDR of its classes is the set of 60×80 matrices

$$G_{6080}(26) = \{ B_{\text{MZO6080}},\, D_{\text{MRVEI6080}} \} \tag{5}$$

by its subset B_{MZO6080} of 2^{4800} MZO and subset $D_{\text{MRVEI6080}}$ of MRVEI, where $\left\{ \mathbf{X}_c = \left(x_{uv}^{\langle c \rangle} \right)_{60 \times 80} \right\}_{c=1}^{26} \subset B_{\text{MZO6080}}$ and M6080I of any EEACL can be as within B_{MZO6080}, as well as within $D_{\text{MRVEI6080}}$. The subset $D_{\text{MRVEI6080}}$ contains SDM6080IPD only. Thus, GT (5) is used for the training process of 2LP, while the classifier on the basis of this 2LP can work on just elements of B_{MZO6080} (grayscale elements of $D_{\text{MRVEI6080}}$ are monochromed by 0.5-crossing comparator, setting a grayscale pixel value to zero or one).

V. Number of Neurons in Perceptron Hidden Layers and MATLAB Function for Training

2LP input layer has 4800 neurons and its output layer has 26 neurons. The size of SHL in 2LP for the problem of classifying SDM6080I can be preset up to 240 neurons. It is an adequate SHL size for GT (5), and such size ensures reasonable speed and accuracy [3], [6], [10], [11], [13]. Then 2LP configuration is 4800-240-26 (input layer neuron number – SHL neuron number – output layer neuron number), and this 4800-240-26 perceptron (4800-240-26-P) is initialised on MATLAB Neural Network Toolbox simply with function "feedforwardnet" or "newff". Function "newff" is preferable for early versions of MATLAB. 4800-240-26-P contains 1158506 weight and bias values.

Perceptrons are trained effectively with a backpropagation algorithm [8], [9], [10], [14], [15], having many methods of its implementation in MATLAB Neural Network Toolbox [8]–[10]. Let us select a MATLAB function "traingda" [8], [9], [10], [16] to train 4800-240-26-P. It is one of the fastest methods of backpropagation algorithm within MATLAB, where weight and bias values are updated according to gradient descent with adaptive learning rate [14], [17], [18].

VI. Model of SDM6080IPD

On k-EFSDM6080I there is M6080I of the c-th class EEACL $\mathbf{X}_c = \left(x_{uv}^{\langle c \rangle} \right)_{60 \times 80}$ to be shifted by SSD (1) horizontally and vertically. Let function $\varphi(\alpha)$ round α to the nearest integer less than or equal to α. Then M6080I is shifted horizontally for

$$\begin{aligned} s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k \rangle}\right) &= \varphi\left(8\sigma_{\text{SDM6080I}}^{\langle k \rangle} \cdot \xi_{\text{hor}}(k)\right) \times \\ &\times \frac{1 - \operatorname{sign}\left(\left| \varphi\left(8\sigma_{\text{SDM6080I}}^{\langle k \rangle} \cdot \xi_{\text{hor}}(k)\right) \right| - 80 \right)}{2} + \\ &+ 80 \cdot \frac{1 + \operatorname{sign}\left(\left| \varphi\left(8\sigma_{\text{SDM6080I}}^{\langle k \rangle} \cdot \xi_{\text{hor}}(k)\right) \right| - 80 \right)}{2} \end{aligned} \tag{6}$$

pixels, where $\xi_{\text{hor}}(k)$ is value of NVZEUV, raffled on k-EFSDM6080I for horizontal pixel shift, and concurrently this M6080I is shifted vertically for

$$s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)=\varphi\left(6\sigma_{\text{SDM6080I}}^{\langle k\rangle}\cdot\xi_{\text{ver}}(k)\right)\times$$
$$\times\frac{1-\operatorname{sign}\left(\left|\varphi\left(6\sigma_{\text{SDM6080I}}^{\langle k\rangle}\cdot\xi_{\text{ver}}(k)\right)\right|-60\right)}{2}+$$
$$+60\cdot\frac{1+\operatorname{sign}\left(\left|\varphi\left(6\sigma_{\text{SDM6080I}}^{\langle k\rangle}\cdot\xi_{\text{ver}}(k)\right)\right|-60\right)}{2} \tag{7}$$

pixels, where $\xi_{\text{ver}}(k)$ is value of NVZEUV, raffled on k-EFSDM6080I for vertical pixel shift. In MATLAB the white colour pixel is coded with 1 and the black colour pixel is coded with 0. And shifting the matrix $\mathbf{X}_c$ horizontally gives the matrix $\tilde{\mathbf{X}}_c(k)=\left[\tilde{x}_{uv}^{\langle c\rangle}(k)\right]_{60\times 80}$, whose elements for $s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)>0$ are

$$\tilde{x}_{uv}^{\langle c\rangle}(k)=1 \quad\text{for}\quad u=\overline{1,\,60} \quad\text{and}\quad v=\overline{1,\,s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)} \tag{8}$$

by

$$\tilde{x}_{uv}^{\langle c\rangle}(k)=x_{ut}^{\langle c\rangle} \quad\text{at}\quad t=v-s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)$$
$$\text{for}\quad u=\overline{1,\,60} \quad\text{and}\quad v=\overline{s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)+1,\,80}\,. \tag{9}$$

Matrix $\mathbf{X}_c$ by $s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)>0$ with (8) and (9) is shifted horizontally to the right. For $s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)<0$ this matrix is shifted horizontally to the left:

$$\tilde{x}_{uv}^{\langle c\rangle}(k)=x_{ut}^{\langle c\rangle} \quad\text{at}\quad t=v-s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)$$
$$\text{for}\quad u=\overline{1,\,60} \quad\text{and}\quad v=\overline{1,\,80+s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)} \tag{10}$$

by

$$\tilde{x}_{uv}^{\langle c\rangle}(k)=1 \quad\text{for}\quad u=\overline{1,\,60} \quad\text{and}$$
$$v=\overline{80+s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)+1,\,80}\,. \tag{11}$$

Clearly, for $s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)=0$ the c-th class NDR is not horizontally shifted:

$$\tilde{x}_{uv}^{\langle c\rangle}(k)=x_{uv}^{\langle c\rangle} \quad\text{for}\quad u=\overline{1,\,60} \quad\text{and}\quad v=\overline{1,\,80}\,. \tag{12}$$

For $s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)>0$ the matrix $\tilde{\mathbf{X}}_c(k)$ is shifted vertically upward, producing the matrix $\tilde{\tilde{\mathbf{X}}}_c(k)=\left[\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k)\right]_{60\times 80}$ of SDM6080I of the c-th class EEACL:

$$\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k)=\tilde{x}_{rv}^{\langle c\rangle}(k) \quad\text{at}\quad r=u+s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)$$
$$\text{for}\quad u=\overline{1,\,60-s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)} \quad\text{and}\quad v=\overline{1,\,80} \tag{13}$$

by

$$\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k)=1 \quad\text{for}$$
$$u=\overline{60-s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)+1,\,60} \quad\text{and}\quad v=\overline{1,\,80}\,. \tag{14}$$

For $s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)<0$ the matrix $\tilde{\mathbf{X}}_c(k)$ is shifted vertically downward:

$$\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k)=1 \quad\text{for}\quad u=\overline{1,\,-s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)} \quad\text{and}\quad v=\overline{1,\,80} \tag{15}$$

by

$$\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k)=\tilde{x}_{rv}^{\langle c\rangle}(k) \quad\text{at}\quad r=u+s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right) \quad\text{for}$$
$$u=\overline{-s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)+1,\,60} \quad\text{and}\quad v=\overline{1,\,80}\,. \tag{16}$$

Clearly, for $s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)=0$ there is no vertical shift:

$$\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k)=\tilde{x}_{uv}^{\langle c\rangle}(k) \quad\text{for}\quad u=\overline{1,\,60} \quad\text{and}\quad v=\overline{1,80}\,. \tag{17}$$

Statements (6)–(17) are a model of making SDM6080I from the c-th class NDR on k-EFSDM6080I [3]. Before making SDM6080IPD, each matrix from matrices $\left\{\tilde{\tilde{\mathbf{X}}}_c(k)\right\}_{c=1}^{26}$ of SDM6080I is reshaped into column vector $\mathbf{Y}_c(k)=\left[y_w^{\langle c\rangle}(k)\right]_{4800\times 1}$,

$$y_w^{\langle c\rangle}(k)=\tilde{\tilde{x}}_{uv}^{\langle c\rangle}(k) \quad\text{by}\quad w=80(u-1)+v$$
$$\text{for}\quad u=\overline{1,\,60} \quad\text{and}\quad v=\overline{1,\,80}\,.$$

Then vectors $\left\{\mathbf{Y}_c(k)\right\}_{c=1}^{26}$ are horizontally concatenated into 4800×26 matrix $\mathbf{Z}_{\text{SDM6080I}}^{\langle k\rangle}=\left[z_{wc}(k)\right]_{4800\times 26}$ whose c-th column is the c-th class representative:

$$z_{wc}(k)=y_w^{\langle c\rangle}(k) \quad\text{for}\quad w=\overline{1,\,4800} \quad\text{by}\quad c=\overline{1,\,26}\,.$$

Therefore, the model of SDM6080IPD is completed with addition [3], [10], [11]

$$\mathbf{Z}_{\text{SDM6080IPD}}^{\langle k\rangle}=\mathbf{Z}_{\text{SDM6080I}}^{\langle k\rangle}+\sigma_{\text{SDM6080IPD}}^{\langle k\rangle}\cdot\mathbf{\Xi} \tag{18}$$

by PDSSD (2) and 4800×26 matrix $\mathbf{\Xi}$ of values of NVZEUV, raffled on k-EFSDM6080IPD, $k=\overline{1,\,F}$.

VII. Range of PSSDR

In the training process by SDM6080IPD the input of

4800-240-26-P is fed with the training set

$$\left\{\left\{\mathbf{X}\right\}_{d=1}^{R}, \left\{\mathbf{Z}_{\text{SDM6080IPD}}^{\langle k\rangle}\right\}_{k=1}^{F}\right\} \tag{19}$$

included $R \in \mathbb{N}$ replicas $\mathbf{X} = \left[x_{wc}\right]_{4800\times 26}$ of 26 NDR $\left\{\mathbf{X}_c = \left(x_{uv}^{\langle c\rangle}\right)_{60\times 80}\right\}_{c=1}^{26}$ by $R+F$ targets as identity 26×26 matrices, where

$$x_{wc} = x_{uv}^{\langle c\rangle} \text{ by } w = 80(u-1)+v$$
$$\text{for } u = \overline{1, 60} \text{ and } v = \overline{1, 80} \text{ by } c = \overline{1, 26}.$$

Set (19) feeds the input of 4800-240-26-P for Q_{pass} times by the said targets. Of course, parameters $\left\{R, F, Q_{\text{pass}}\right\}$ of 4800-240-26-P are better to be heuristically adjusted for evaluating the function $v_{\text{CEP}}(r)$ and solving problem (4) as fast as possible, not losing accuracy at that. Here appropriately to put

$$\left\{R, F, Q_{\text{pass}}\right\} = \{1, 4, 80\} \tag{20}$$

into 4800-240-26-P. Triplet (20) provides enough accurate and fast evaluation [3]. But range $\left[r_{\min}; r_{\max}\right]$ of PSSDR should not be overextended to prevent delays in evaluation procedures.

In PSSDR (3) the variable is $\sigma_{\text{SDM6080IPD}}^{\langle\max\rangle}$ while the maximal SSD is constant. Integers $\left|s_{\text{hor}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)\right|$ and $\left|s_{\text{ver}}\left(\sigma_{\text{SDM6080I}}^{\langle k\rangle}\right)\right|$ must not be very great for M6080I of NDR, so EEACL shall not be outside the contour of M6080I, and its larger part shall be within the contour. For this, it is acceptable to put $\sigma_{\text{SDM6080I}}^{\langle\max\rangle} = 1$. From the experience if $\sigma_{\text{SDM6080IPD}}^{\langle\max\rangle} > 2$ then 4800-240-26-P by its parameters (20) is trained more for SDM6080IPD classification, rather than for SDM6080I. Besides, the trained at $r > 2$ 4800-240-26-P performs with inadmissibly great CEP. Hence put $r_{\max} = 2$. And $r_{\min} = 0.025$ as at lesser PSSDR the training process of 4800-240-26-P may drag for nought. Thus, the range $[0.025; 2]$ of PSSDR has been determined.

VIII. Running through the Range of PSSDR

When 4800-240-26-P has been trained by its parameters (20) and fixed PSSDR (3), it is tested under some SSD $\sigma_{\text{SDM6080I}} \in \left[0; \sigma_{\text{SDM6080I}}^{\langle\max\rangle}\right] = [0; 1]$. In this way CEP $v_{\text{CEP}}\left(r, \sigma_{\text{SDM6080I}}\right)$ is an $\left\{r, \sigma_{\text{SDM6080I}}\right\}$-performance of 4800-240-26-P, and its average over the segment $\left[0; \sigma_{\text{SDM6080I}}^{\langle\max\rangle}\right]$ of the testing SSD is

$$v_{\text{CEP}}(r) = \frac{1}{\sigma_{\text{SDM6080I}}^{\langle\max\rangle}} \int_0^{\sigma_{\text{SDM6080I}}^{\langle\max\rangle}} v_{\text{CEP}}\left(r, \sigma_{\text{SDM6080I}}\right) d\sigma_{\text{SDM6080I}} =$$

$$= \int_0^1 v_{\text{CEP}}\left(r, \sigma_{\text{SDM6080I}}\right) d\sigma_{\text{SDM6080I}} \tag{21}$$

by PSSDR $r \in [0.025; 2]$. An $\left\{r, \sigma_{\text{SDM6080I}}\right\}$-performance is statistically calculated after 400 batches of 26 SDM6080I (where every class is represented) have been run through 4800-240-26-P. On account of that 4800-240-26-P will not be tested for a short period, there is the sampled subset

$$\{0.1i\}_{i=0}^{10} \subset \left[0; \sigma_{\text{SDM6080I}}^{\langle\max\rangle}\right] = [0; 1] \tag{22}$$

for covering the range of the testing SSD. On subset (22) the integral in (21) can be approximated numerically:

$$v_{\text{CEP}}(r) \approx \frac{1}{11}\sum_{i=0}^{10} v_{\text{CEP}}(r, 0.1i). \tag{23}$$

The range of PSSDR is sampled rough: there is PSSDR segment subset

$$\left\{\{0.025i\}_{i=1}^{4}, \{0.1i\}_{i=2}^{20}\right\} \subset \left[r_{\min}; r_{\max}\right] = [0.025; 2] \tag{24}$$

and for each of 23 points in (24) value (23) is calculated (Fig. 1).

Obviously, Fig. 1 leaves rough notion about the minimum of the function $v_{\text{CEP}}(r)$. The point $r = 0.075$ is hardly seen as the minimum. And the point $r = 0.025$ being here the closest one to absence of pixel distortion looks suspiciously low (although its TPD is the longest). The peak at $r = 0.05$ is queer. Nevertheless, it is clear that the problem (4) should be re-stated into

$$r^* \in \arg\min_{r\in(0;\,2]} v_{\text{CEP}}(r) \tag{25}$$

and it seems the optimal PSSDR $r^* \in (0; 0.2)$.

Figure 2 contains additional evaluations of the function $v_{\text{CEP}}(r)$ on sampled subsegments

$$\{0, 0.001, 0.002\},\ \{0.0025, 0.005, 0.0075, 0.01, 0.02\},$$
$$\{0.025, 0.05, 0.075, 0.1, 0.2\} \tag{26}$$

of the segment $[0; 0.2]$ involved 100 trained 4800-240-26-P by parameters (20) for each of the sets (26). By importing the 30 trained 4800-240-26-P off Fig. 1, the re-evaluation of the function $v_{\text{CEP}}(r)$ in Fig. 3 drops impression of that those peaks (not excluding the mentioned above at $r = 0.05$) truly exist. In fact, the peak at $r = 0.05$ became stronger (higher), and the minimum at $r = 0.075$ (Fig. 3) was re-registered.

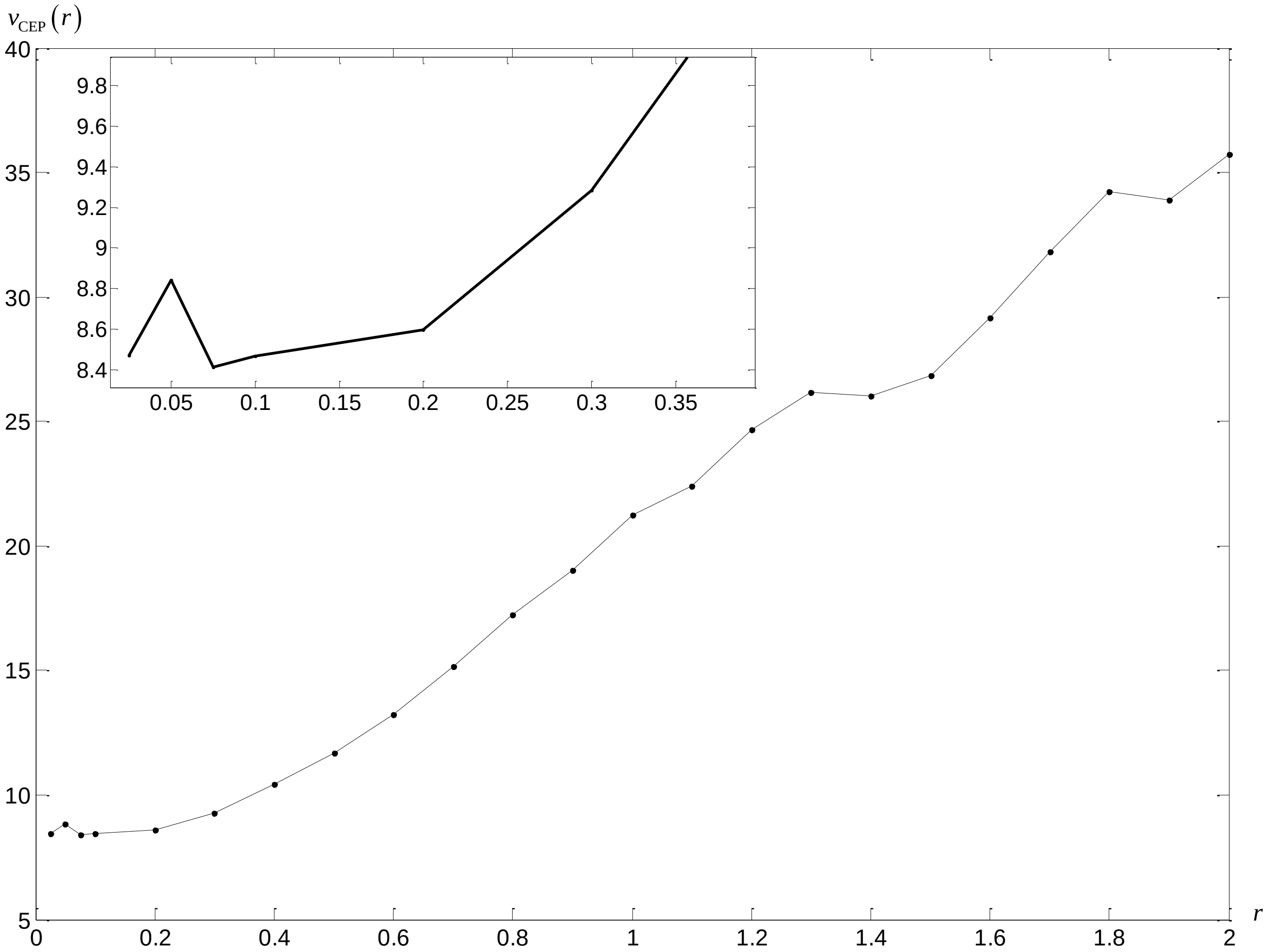

Fig. 1. Evaluation of the function $v_{\text{CEP}}(r)$ on 30 trained 4800-240-26-P by parameters (20).

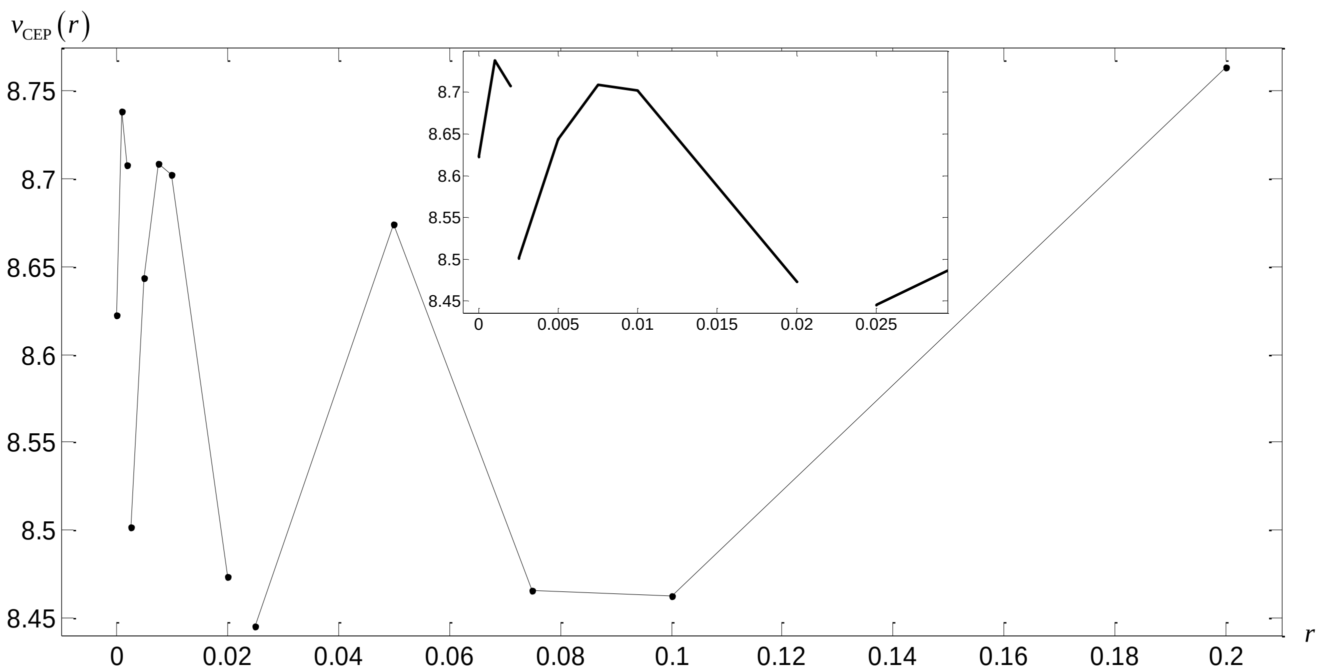

Fig. 2. Additional evaluations of the function $v_{\text{CEP}}(r)$ on 100 trained 4800-240-26-P by parameters (20) for each of the three sets (26).

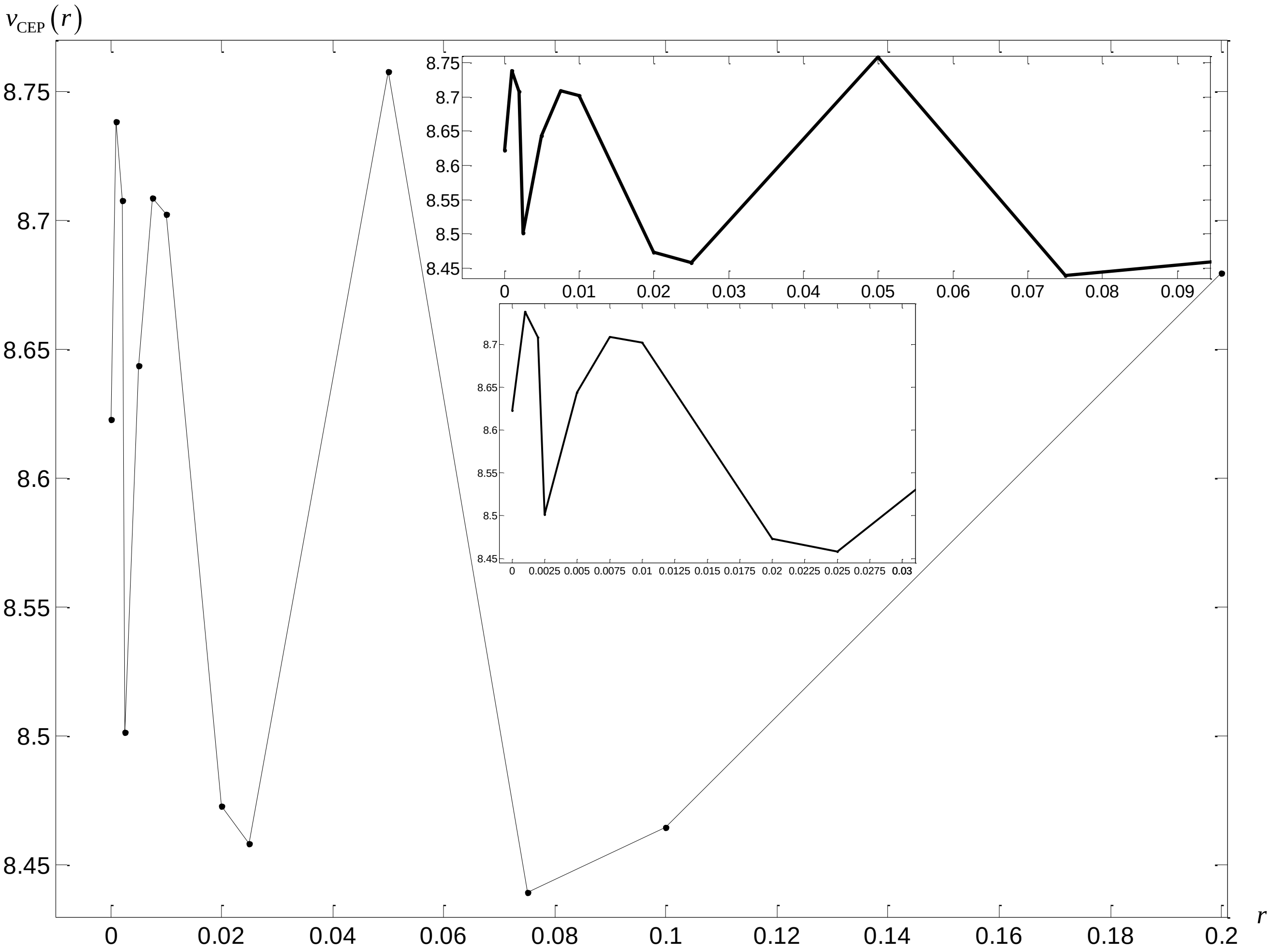

Fig. 3. Re-evaluation of the function $v_{\text{CEP}}(r)$ on 100 trained 4800-240-26-P by parameters (20) for the first and second sets in (26); the third set in (26) is evaluated on 130 trained 4800-240-26-P.

No matter that the point $r = 0.075$ appears the minimum again. The point $r = 0.025$ pretends to be minimum as well. And any conclusion on a minimum interval would have been inconsistent because of those three peaks. So let us count the re-evaluation (Fig. 3) under parameters (20) to be a fast-and-rough approximation in solving the problem (25). Fulfilling the training process under the strengthened parameters

$$\{R, F, Q_{\text{pass}}\} = \{2, 8, 240\} \tag{27}$$

could make the evaluation more accurate, because the classifier performance in Fig. 3 is not good enough. It directs to solving the problem

$$r^* \in \arg\min_{r \in (0;\, 0.5]} v_{\text{CEP}}(r) \tag{28}$$

by extending the right endpoint to preserve possible minima outside of the half-segment

$$(0;\, 0.2] \subset (0;\, 0.5] \subset (0;\, 2].$$

IX. The Averaged CEP Minimisation due to the Problem (25)

Having re-sampled the segment $[0;\, 0.5]$ as

$$\left\{ \{0.025i\}_{i=0}^{4},\, \{0.1 + 0.05i\}_{i=1}^{8} \right\} \subset [0;\, 0.5] \tag{29}$$

and run 4800-240-26-P by its parameters (27) through 13 points (29), there is more accurate solution (Fig. 4) for the problem (28): $r^* = 0.05$ with 0.025 PSSDR axis accuracy. Undoubtedly, increasing PSSDR axis accuracy up to 0.005 and higher leads to slightly another point r^*, but statistically this fails. That is why the minimum

$$v_{\text{CEP}}(r^*) = v_{\text{CEP}}(0.05) \approx 4.0894 \tag{30}$$

of the averaged CEP found in Fig. 4 will not be refined. The best 4800-240-26-P here has been trained namely by PSSDR $r^* = 0.05$, and its averaged CEP

$$v_{\text{CEP}}(r^*) = v_{\text{CEP}}(0.05) < 2.8278 \tag{31}$$

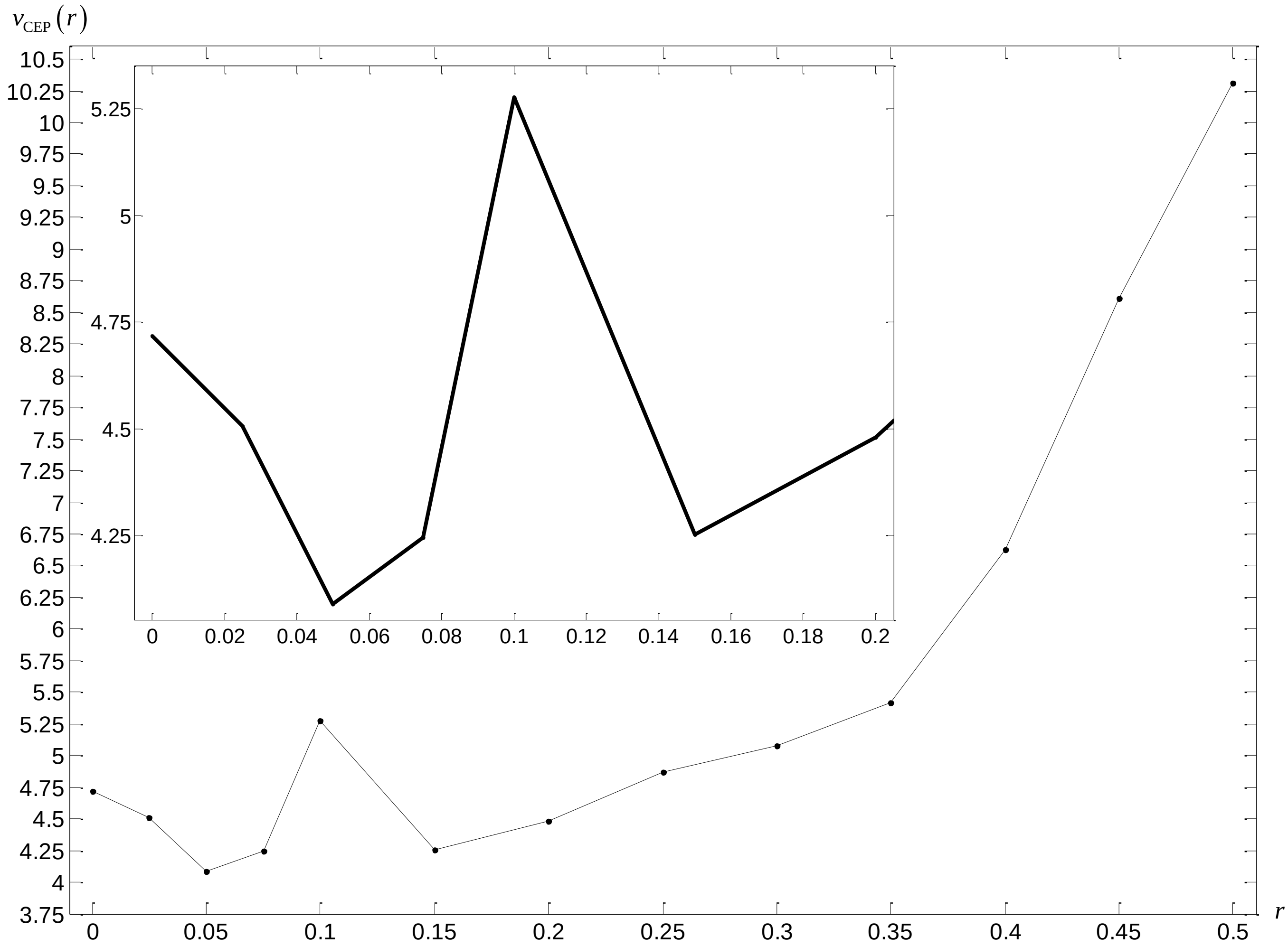

Fig. 4. Evaluation of the function $v_{\mathrm{CEP}}(r)$ on 25 trained 4800-240-26-P by parameters (27).

and

$$v_{\mathrm{CEP}}\left(r^*, \sigma_{\mathrm{SDM6080I}}^{\langle\max\rangle}\right) = v_{\mathrm{CEP}}(0.05, 1) < 13.45, \quad (32)$$

giving confusion for nearly two SDO among 15 EEACL at the maximal shift distortion. The minimum $r^* = 0.05$ is also seen in Fig. 5, where all 25 polylines making the evaluation in Fig. 4 are shown. The peaks at $r = 0.025$ and $r = 0.1$ are prominent, while all 25 4800-240-26-P by $r^* = 0.05$ do not have CEP scattering. The scattering at $r = 0.1$ looks systematic, and at $r = 0$ the scattering is high.

Setting $r > 0$ shortens TPD (Fig. 6) that diminishes any doubts as to whether positive PSSDR is needed. The minimal averaged TPD, however, is not at $r^* = 0.05$, but at $r = 0.3$, which is not surprising. And starting right after $r > 0.3$ TPD increases (the cause is normal noise overload).

Further it remains just to verify that by PSSDR $r^* = 0.05$ the classifier 4800-240-26-P really recognises SDM6080I with the minimal CEP. But based on the best 4800-240-26-P with its (31) and (32), the classifier still can be optimized via additional training. For extra $Q_{\mathrm{pass}} = 18$, the optimized 4800-240-26-P (4800-240-26-PO) has

$$v_{\mathrm{CEP}}\left(r^*\right) = v_{\mathrm{CEP}}(0.05) < 2.5 \quad (33)$$

and

$$v_{\mathrm{CEP}}\left(r^*, \sigma_{\mathrm{SDM6080I}}^{\langle\max\rangle}\right) = v_{\mathrm{CEP}}(0.05, 1) < 12.18, \quad (34)$$

leaving behind other 2LP classifiers. Figure 7 presents what those shifts are by SSD $\sigma_{\mathrm{SDM6080I}} = \sigma_{\mathrm{SDM6080I}}^{\langle\max\rangle} = 1$ and that SDM6080IPD-trained classifier 4800-240-26-PO by PSSDR $r^* = 0.05$ performs over such SDM6080I finely, erring only in one case from eight takings. Figure 8 with the functions $v_{\mathrm{CEP}}(0.05, \sigma_{\mathrm{SDM6080I}})$ and $v_{\mathrm{CEP}}(0.25, \sigma_{\mathrm{SDM6080I}})$ on SSD segment $\left[0; \sigma_{\mathrm{SDM6080I}}^{\langle\max\rangle}\right] = [0; 1]$ proves that the classifier 4800-240-26-PO performs with smaller CEP through the whole range of SSD.

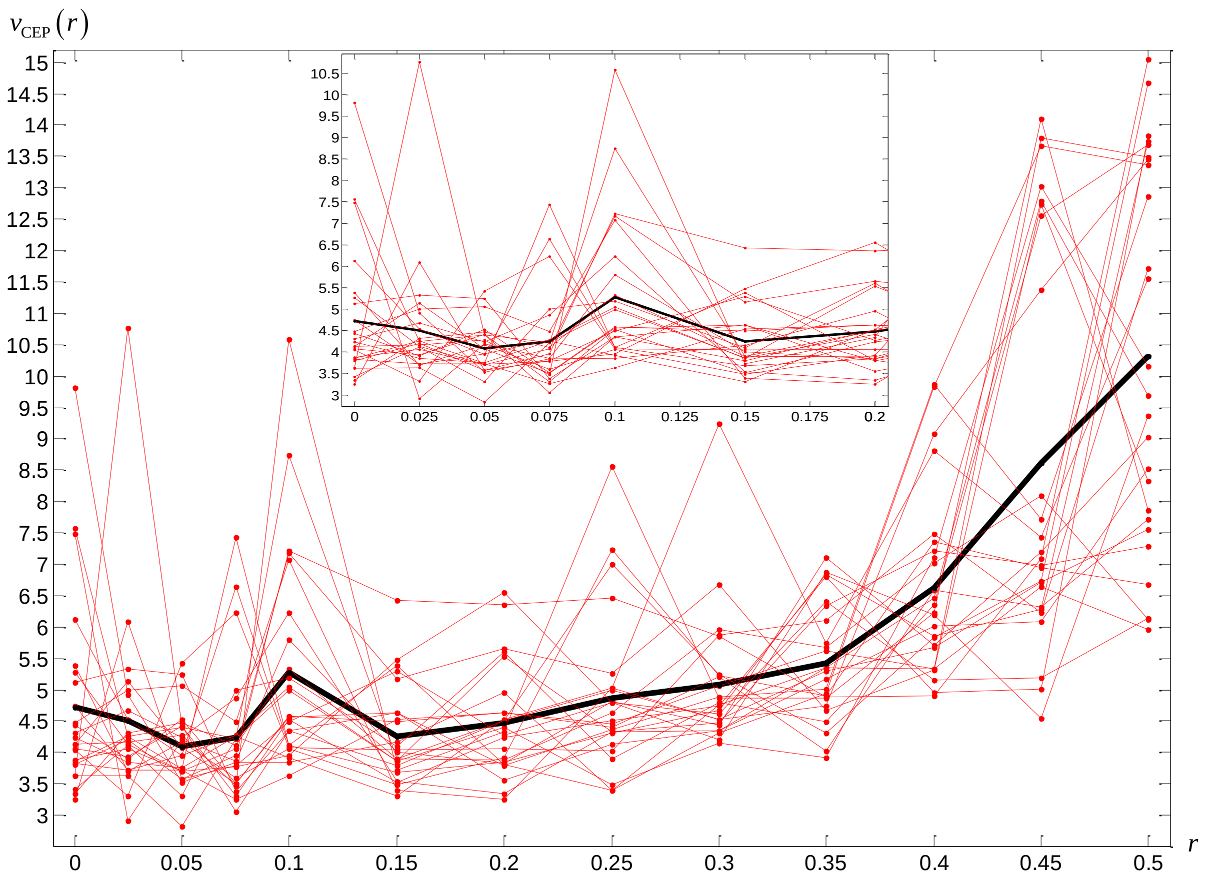

Fig. 5. All 25 polylines making the evaluation in Fig. 4 (the evaluation is drawn with solid line).

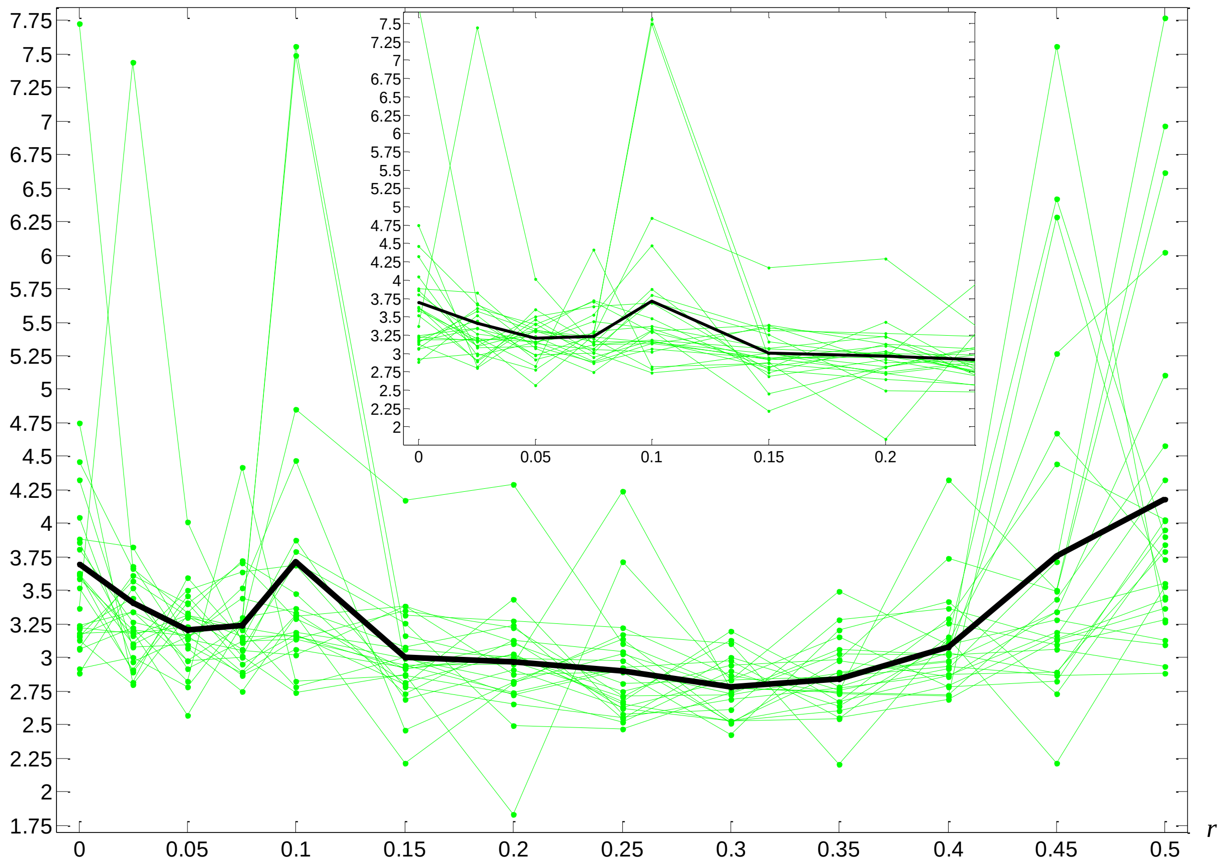

Fig. 6. Relative TPD for those 25 polylines making the evaluation in Fig. 4 (the averaged TPD polyline is solid).

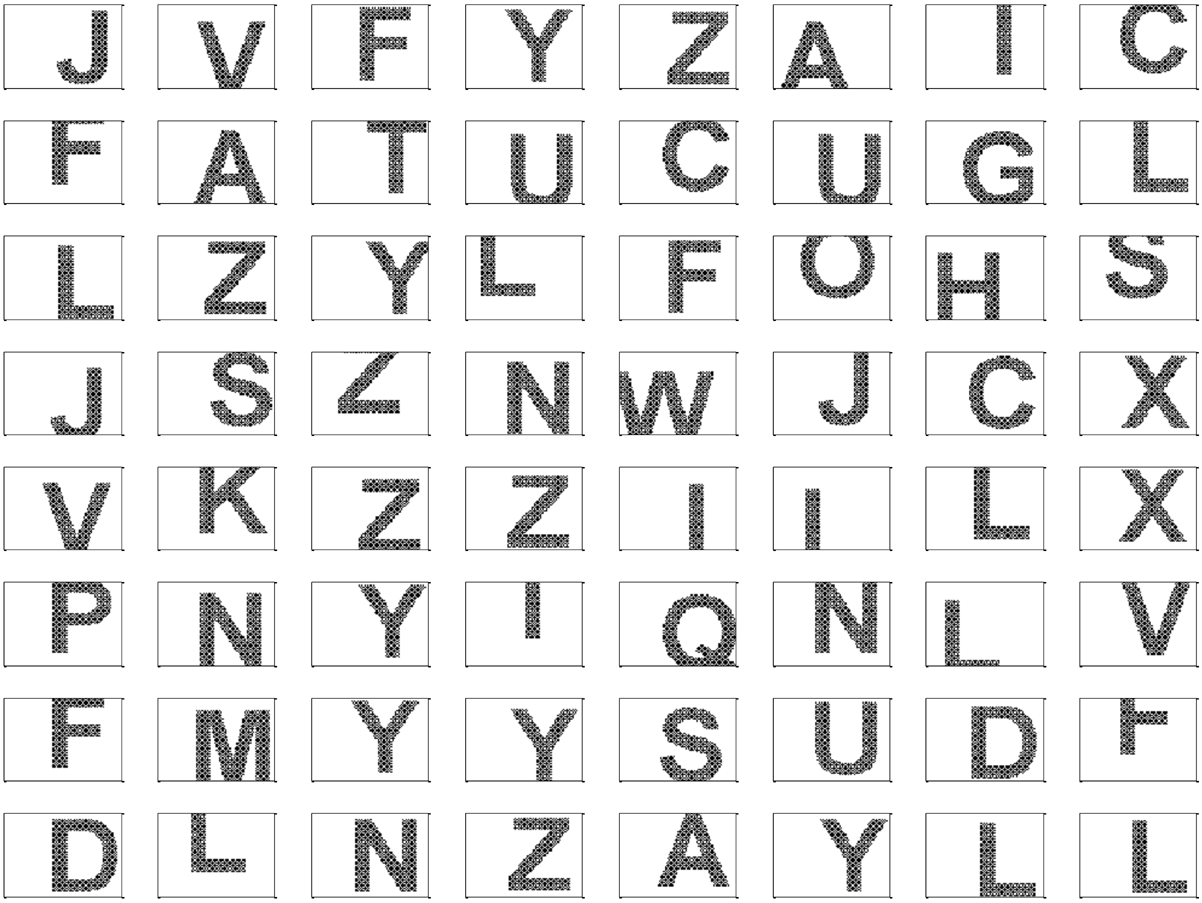

Fig. 7. SDM6080I by SSD $\sigma_{\mathrm{SDM6080I}} = 1$, recognised with SDM6080IPD-trained classifier 4800-240-26-PO by PSSDR $r^{*} = 0.05$ at CEP (34).

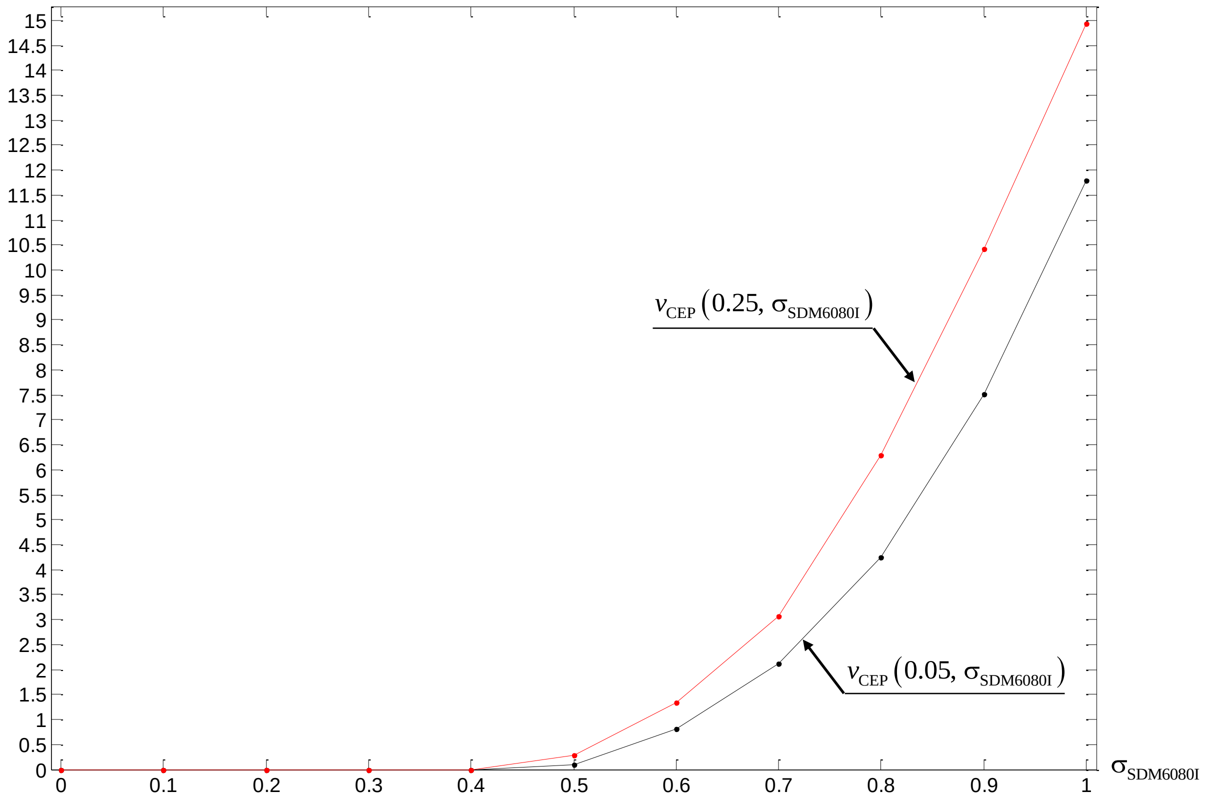

Fig. 8. Evaluation of the function $v_{\mathrm{CEP}}(0.05,\ \sigma_{\mathrm{SDM6080I}})$ for 4800-240-26-PO and evaluation of the function $v_{\mathrm{CEP}}(0.25,\ \sigma_{\mathrm{SDM6080I}})$ for the best 2LP from [3].

Now, the averaged CEP for 4800-240-26-P has been minimised due to the problem (25). PSSDR $r^* = 0.05$ allows effectively classifying SDM6080I on the pattern of GT (5). It is self-evident that for SDO of other types (for instance, images of other formats) the found optimal value of PSSDR will be different.

X. Conclusion

The suggested routine allowed using optimal PSSDR for SDM6080I to decrease CEP by 20 % in comparison with the blindly fixed PSSDR (see Fig. 8). Resolving other problems of SDO classification, the optimal value of PSSDR can be found with an item-by-item approach stated in the section of the paper aim and task formulation. Those six item (task) line-up is not fully universal, where the first and third items are not general. So they may be re-stated into the following:

1. To define NDR and number of classes in GT that contains SDO, which will feed the input of the classifier. To isolate GT, containing SDO and NNO, which will be used for the training process.
2. To configure the classifier architecture and select the program environment, where this architecture is going to be modelled and simulated.
3. To put statements for the model of SDO and for the model of SDO additionally distorted with NNO.
4. To determine a narrow range $[r_{\min}; r_{\max}]$ of PSSDR, where the minimum point of the function $v_{\text{CEP}}(r)$ shall be confidently enclosed.
5. To run the classifier through the sampled range of PSSDR in order to evaluate the function $v_{\text{CEP}}(r)$. If the minimum point is found roughly then a subrange within the range $[r_{\min};r_{\max}]$, confidently enclosed the minimum, ought to be re-sampled and the classifier must be re-run again through the re-sampled subrange. This loops until the minimum point of the function $v_{\text{CEP}}(r)$ is found with admissible accuracy.
6. To validate that the identified classifier under the found optimal PSSDR value performs with the minimized CEP over SDO on the pattern of the defined GT.

Optimization of 2LP in classifying SDO shall be continued. This is essential because parameters (27) were put empirically rather than after optimization procedure or close to that. Besides, the size of SHL in 2LP also requires to be substantiated (maybe, 239 neurons in SHL would have shortened TPD or 241 neurons in SHL would have lowered CEP) and, moreover, k-EFSDM6080IPD in (18) might have been updated during the training process. These various optimization mathematical problems, however, may be solved only on a pattern, not on abstract objects and their model.

References

[1] K. Fukushima and S. Miyake, “Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position,” *Pattern Recognition*, vol. 15, iss. 6, 1982, pp. 455–469. http://dx.doi.org/10.1016/0031-3203(82)90024-3

[2] K. Fukushima, “Self-organization of shift-invariant receptive fields,” *Neural Networks*, vol. 12, iss. 6, 1999, pp. 791–801.

[3] V. V. Romanuke, “An attempt for 2-layer perceptron high performance in classifying shifted monochrome 60-by-80-images via training with pixel-distorted shifted images on the pattern of 26 alphabet letters,” *Radioelectronics, informatics, control*, no. 2, 2013, pp. 112–118. http://dx.doi.org/10.15588/1607-3274-2013-2-18

[4] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position,” *Biological Cybernetics*, vol. 36, iss. 4, 1980, pp. 193–202.

[5] K. Fukushima, “Training multi-layered neural network neocognitron,” *Neural Networks*, vol. 40, 2013, pp. 18–31. http://dx.doi.org/10.1016/j.neunet.2013.01.001

[6] G. Arulampalam and A. Bouzerdoum, “A generalized feedforward neural network architecture for classification and regression,” *Neural Networks*, vol. 16, iss. 5–6, 2003, pp. 561–568. http://dx.doi.org/10.1016/S0893-6080(03)00116-3

[7] C. Yu, M. T. Manry, J. Li, and P. L. Narasimha, “An efficient hidden layer training method for the multilayer perceptron,” *Neurocomputing*, vol. 70, iss. 1–3, 2006, pp. 525–535. http://dx.doi.org/10.1016/j.neucom.2005.11.008

[8] V. V. Romanuke, “Setting the hidden layer neuron number in feedforward neural network for an image recognition problem under Gaussian noise of distortion,” *Computer and Information Science*, vol. 6, no. 2, 2013, pp. 38–54. http://dx.doi.org/10.5539/cis.v6n2p38

[9] V. V. Romanuke, “Dependence of performance of feed-forward neuronet with single hidden layer of neurons against its training smoothness on noised replicas of pattern alphabet,” Bulletin of Khmelnitskiy National University. *Technical Sciences*, no. 1, 2013, pp. 201–206.

[10] 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,” *Research bulletin of the National Technical University of Ukraine “KPI”*, no. 5, 2014, pp. 55–62.

[11] V. V. Romanuke, “Classification error percentage decrement of two-layer perceptron for classifying scaled objects on the pattern of monochrome 60-by-80-images of 26 alphabet letters by training with pixel-distorted scaled images,” Scientific bulletin of Chernivtsi National University of Yuriy Fedkovych. *Series: Computer systems and components*, vol. 4, iss. 3, 2013, pp. 53–64.

[12] P. A. Castillo, J. J. Merelo, M. G. Arenas, and G. Romero, “Comparing evolutionary hybrid systems for design and optimization of multilayer perceptron structure along training parameters,” *Information Sciences*, vol. 177, iss. 14, 2007, pp. 2884–2905. http://dx.doi.org/10.1016/j.ins.2007.02.021

[13] S. S. Malalur, M. T. Manry, and P. Jesudhas, “Multiple optimal learning factors for the multi-layer perceptron,” *Neurocomputing*, vol. 149, p. C, 2015, pp. 1490–1501. http://dx.doi.org/10.1016/j.neucom.2014.08.043

[14] M. T. Hagan and M. B. Menhaj, “Training feedforward networks with the Marquardt algorithm,” *IEEE Transactions on Neural Networks*, vol. 5, iss. 6, 1994, pp. 989–993. http://dx.doi.org/10.1109/72.329697

[15] T. Kathirvalavakumar and S. Jeyaseeli Subavathi, “Neighborhood based modified backpropagation algorithm using adaptive learning parameters for training feedforward neural networks,” *Neurocomputing*, vol. 72, iss. 16–18, 2009, pp. 3915–3921. http://dx.doi.org/10.1016/j.neucom.2009.04.010

[16] V. V. Romanuke, “Accuracy improvement in wear state discontinuous tracking model regarding statistical data inaccuracies and shifts with boosting mini-ensemble of two-layer perceptrons,” *Problems of tribology*, no. 4, 2014, pp. 55–58.

[17] A. Nied, S. I. Jr. Seleme, G. G. Parma, and B. R. Menezes, “On-line neural training algorithm with sliding mode control and adaptive learning rate,” *Neurocomputing*, vol. 70, iss. 16–18, 2007, pp. 2687–2691. http://dx.doi.org/10.1016/j.neucom.2006.07.019

[18] S. J. Yoo, J. B. Park, and Y. H. Choi, “Indirect adaptive control of nonlinear dynamic systems using self recurrent wavelet neural networks via adaptive learning rates,” Information Sciences, vol. 177, iss. 15, 2007, pp. 3074–3098. http://dx.doi.org/10.1016/j.ins.2007.02.009

Vadim V. Romanuke was born in 1979. Higher education was received in 2001. In 2006, he received the Degree of Candidate of Technical Sciences in Mathematical Modelling and Computational Methods. The degree of Doctor of Technical Sciences in Mathematical Modelling and Computational Methods was received in 2014.

He is a Professor of the Department of Applied Mathematics and Social Informatics at Khmelnitskiy National University. His research interests concern decision making, game theory, statistical approximation, and control engineering based on statistical correspondence.

Address: Institutskaya Str., 11, 29016, Khmelnitskiy, Ukraine.

E-mail: romanukevadimv@mail.ru