Journal of Uncertain Systems
Vol.9, No.4, pp.286-305, 2015
Online at: www.jus.org.uk

Two-Layer Perceptron for Classifying Flat Scaled-Turned-Shifted Objects by Additional Feature Distortions in Training

Vadim V. Romanuke*
Department of Applied Mathematics and Social Informatics, Khmelnitskiy National University, Institutskaya str., 11, 29016, Khmelnitskiy,Ukraine

Received 25 December 2013; Revised 2 August 2015

Abstract

A problem of classifying diversely distorted objects is considered. These objects are modeled as flat scaled-turned-shifted objects. In order to substitute complicated and slow deep learning architectures, two-layer perceptron is tried to be a classifier. For this, the perceptron classifier is trained by a specific embedding into the training process. The embedding is additional feature distortions regularizing the training so that it makes the perceptron classify diversely distorted objects more accurately. In classifying 60×80 monochrome images of 26 enlarged capital letters, an appropriately trained perceptron is identified which performs with less than 4% of errors at medium intensity of distortions. The perceptron keeps its high-accurate classification capabilities if total number of object features and classes' number are different from those 4800 and 26, respectively. Furthermore, unlike deep learning classifiers, two-layer perceptron can take just about half a minute to classify 10000 diversely distorted objects.

Keywords: classification of diversely distorted objects, deep learning classifiers, two-layer perceptron, training set, classification error percentage

1 Introduction

Object classification is a great part of global automatization problem. Classifiers are required to be light, simple, and rapid. In classification of diversely distorted objects (DDO), the classifier's simplicity and rapidity are contrary ones standing against each other. Typically, DDO are 2D or 3D data which usually can be imaged. DDO constitute majority among static objects to be classified.

For the present-day paradigm, DDO are classified accurately just by deep learning classifiers (DLC). The high accuracy of DLC is ensured due to that they are based on sets of algorithms that attempt to model high-level abstractions in data by using model architectures, with complex structures or otherwise, composed of multiple nonlinear transformations [39, 28]. Various deep learning architectures such as deep neural networks, convolutional deep neural networks, deep belief networks and recurrent neural networks have been successfully applied to DDO classification [10, 51, 19, 37, 4]. Particularly, neocognitron [14, 15] performing perfectly over 2D objects distorted in diverse ways [18, 19], is used for optical and handwritten character recognition [16, 17]. Cresceptron being a cascade of many layers similar to neocognitron is used for performing 3D object recognition directly from cluttered scenes [47].

However, DLC is too complicated to be rapid over large-scale data [13, 32, 41]. And their training lasts too long as well [20, 32]. Contrariwise, two-layer perceptron (TLP) is much lighter, and classifiers of this type are fast enough [23, 2]. But TLP has, commonly, low-accuracy performance over DDO. Thus, highly accurate DLC stand against rapid TLP classifiers. Nevertheless, this is not only accuracy against rapidity. Here also heavily complicated architecture of DLC is contrasted with simplicity of TLP. So, the motivation is to identify the TLP classifier whose accuracy over DDO would be nearly comparable to high-accuracy performance of DLC so that accuracy, rapidity, and simplicity coalesce. This seems realizable according to the known universal approximation theorem for neural networks (UATNN) [5, 6, 25, 40, 12, 8, 48, 44, 30]. The UATNN states that every continuous function that maps

* Corresponding author.
Email: romanukevadimv@mail.ru (V.V. Romanuke).

intervals of real numbers to some output interval of real numbers can be approximated arbitrarily closely by a TLP. This result holds for a wide range of activation functions, e. g. for the sigmoid functions [11, 1, 9].

2 Antecedent Problems of Static DDO Classification

Preprocessing, a special procedure [35, 43, 3, 24], precedes the static DDO classification. This procedure aims at reduction of redundant object features — constant data, borders, repeating fragments, etc. Another problem, standing before classification, is segmentation of a field of those objects [49, 45, 31] to be classified. The segmentation is accomplished by a criterion aiming at selecting the objects within the field rationally, without regions not relating to those objects [29, 46]. If preprocessing and segmentation are accomplished inaccurately, a DLC is constructed more complicated and retarding [19, 20, 47, 10, 39, 42]. TLP is never dependent upon these antecedent problems of static DDO classification. It's an argument in favor of classifying DDO by TLP.

3 Restrictions of DLC in Static DDO Classification

Further TLP favoring argumentation comes out from that operation speed of DLC is low and depends stronger on the total number of object features (TNOF). DLC don't work on smaller hard disk space and memory [39, 47, 10, 42, 20]. And as TNOF increases, DLC slow down distinctly [4, 19, 47, 34, 51].

These restrictions of DLC in static DDO classification don't concern TLP. For classifying DDO with acceptable accuracy, TLP needs special training process [21, 50]. Only after TLP is successfully trained [50, 27, 36], its low resources consumption, independence upon antecedent problems of static DDO classification, and rapidity are making sense.

4 Goal and Tasks

When the TLP hidden layer neuron number is assigned appropriately, classification capabilities of TLP are fully determined with the TLP training process [23, 7, 21, 27, 22]. Therefore, the TLP training process is to be adjusted and optimized for getting its maximal performance in classifying DDO.

For presenting a DDO deeper, we need to try a few types of distortion simultaneously. Owing to that distortions (deformations) of a flat image are well-visible and best-comprehensible, the flat monochrome image (FMI) is an eminently suitable model of the object.

There are five types of FMI distortion: pixel noise, pixel inversion, scaling, turning (rotation or angulation), and shift. These types are cited in order of complexity in training the TLP. Pixel noise (inversion) distortion is surmounted successfully by the trained TLP [23, 38]. Scaled, turned, or shifted FMI by the greater TNOF are classified much worse. Any of these distortion types requires a specific embedding into the TLP training process. The embedding shall relate to the corresponding distortion type.

The final goal is to identify the TLP classifier which would be capable to substitute DLC in classifying DDO. Flat scaled-turned-shifted objects (FSTSO) will emulate those DDO. And FSTSO are scaled-turned-shifted monochrome images (STSMI). They may be scaled-turned-shifted in diverse ratios and ways.

We are to reach the goal in the following steps:

1. The general totality of objects is defined. TNOF shall be within the medium range.

2. Models of DDO as FSTSO are stated. These models are pixel-distorted monochrome images (PDMI), STSMI, and pixel-distorted STSMI (PDSTSMI).

3. TLP structure is stated.

4. PDMI-trained TLP is checked once again that it cannot classify STSMI or PDSTSMI.

5. TLP is checked out whether it can be trained with STSMI. If there is a fail in this, then TLP is going to be trained with PDSTSMI. And the part of STSMI within the training set of PDSTSMI is increased until the near-minimal classification error percentage (CEP) in classifying STSMI is maintained. In this way, we are to make gradual transition from PDMI-trained TLP to PDSTSMI-trained TLP.

6. Conclusion to effectiveness of PDSTSMI-trained TLP for classifying STSMI is to be made. The effectiveness is understood that the identified TLP classifier is capable to substitute DLC in classifying FSTSO which model DDO. Notices about peculiar training process and specific ratio for types of distortion are to be included.

5 General Totality of Objects to be Classified

The medium range of TNOF implies a few thousands of those features, at which DLC are pretty complicated and slow. At the same time, classification results are naturally desired to be obtained both as fast as possible and appropriate. Also number of classes in the general totality must be moderate. While training and testing, a medium TNOF and a moderate number of classes prevent lingering the investigation processes. Therefore, let FMI be the enlarged English alphabet capital letter of the format 60×80, making up 26 classes. And let FMI be presented with 60×80 matrix of ones and zeros. Thus, the general totality of 60×80 matrices of ones and zeros is constituted from 2^{4800} FMI, where STSMI exist as well.

6 Models of DDO

In the general case, monochrome $Y\times W$ image is presented as a $Y\times W$ matrix. For $Q\in\mathbb{N}\setminus\{1\}$ classes, let $\mathbf{A}_q=\left(a_{uv}^{\langle q\rangle}\right)_{Y\times W}$ be the matrix of elements $a_{uv}^{\langle q\rangle}\in\{0,1\}$ by $q=\overline{1,Q}$ that maps the non-distorted representative of the q-th class. Henceforward any class representative as a $Y\times W$ matrix is reshaped into $(Y\cdot W)$-length-column. Our investigations are going to be carried out from MATLAB environment, wherein the white color is coded with ones, and the black color is coded with zeros (Figure 1).

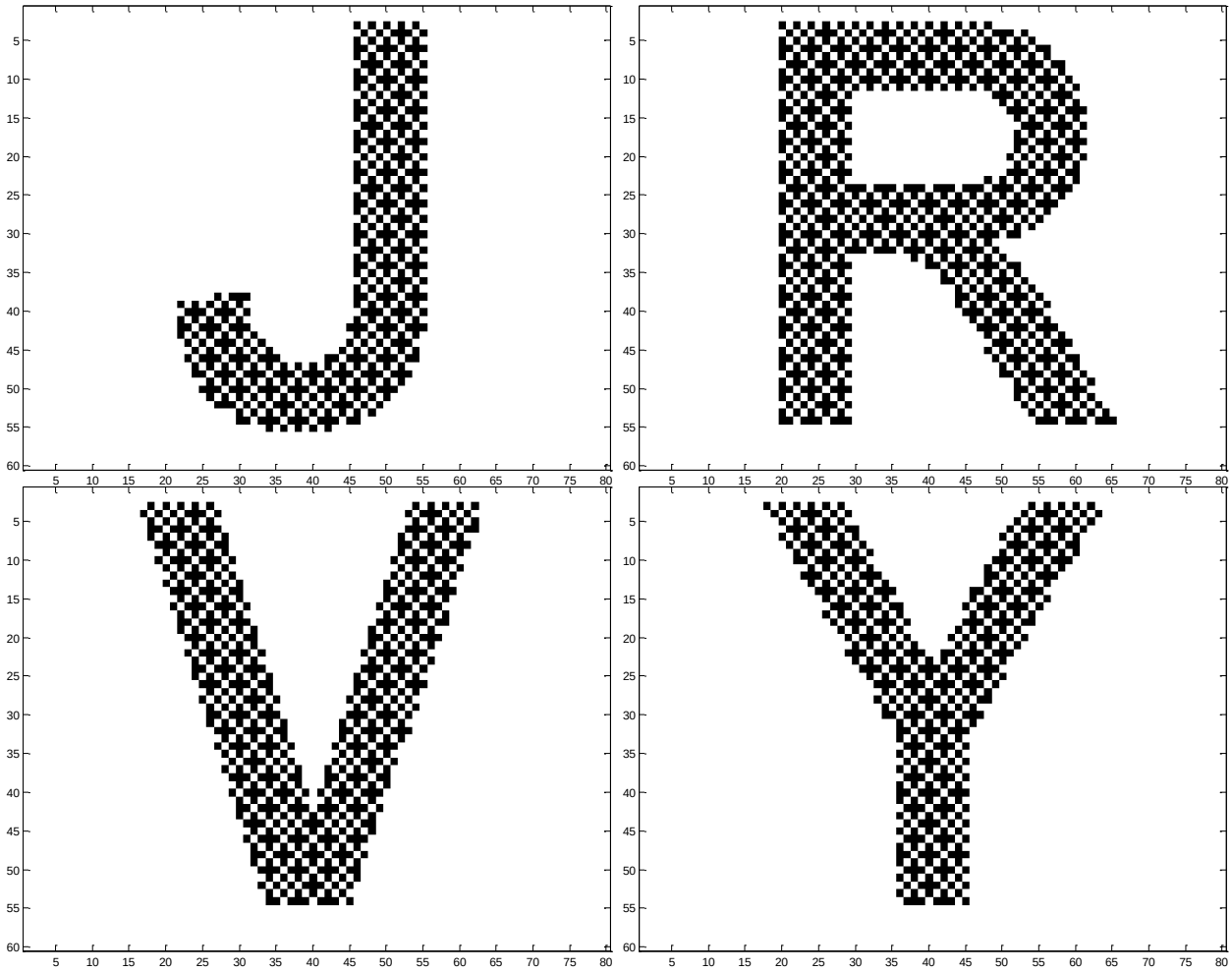

Figure 1: Non-distorted 60×80 FMI of the letters "J", "R", "V", "Y", based on the respective 60×80 matrices $\{\mathbf{A}_{10}, \mathbf{A}_{18}, \mathbf{A}_{22}, \mathbf{A}_{25}\}$ and viewed from within MATLAB

An interesting fact is that FMI is not the continuous black cast on the continuous white background. Regular crosshatching pixel black-into-white inversions throughout the letter cast are seen in Figure 1. Thus, the initial pixel inversion distortion (binary feature inversion distortion) is already put into non-distorted representatives.

6.1 Model of PDMI

For the PDMI training process, all the representatives as $(Y \cdot W)$-length-columns are concatenated horizontally. The Q non-distorted representatives are concatenated into the matrix $\mathbf{A} = \left(\overline{a}_{jq}\right)_{(Y \cdot W) \times Q}$, where its q-th column is the column-reshaped representative of the q-th class. The model of PDMI is in forming the matrix $\mathbf{A}_{\text{PDMI}}^{\langle k \rangle} = \left(\overline{a}_{jq}^{\langle k\text{-PDMI} \rangle}\right)_{(Y \cdot W) \times Q}$ of Q pixel-distorted representatives (each for its class) through the addition

$$\mathbf{A}_{\text{PDMI}}^{\langle k \rangle} = \mathbf{A} + \sigma_{\text{pixel}}^{\langle k \rangle} \cdot \mathbf{\Xi} \tag{1}$$

by standard deviation (SD)

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

and its maximum $\sigma_{\text{pixel}}^{\langle \max \rangle} > 0$ at $(Y \cdot W) \times Q$ matrix $\mathbf{\Xi}$ of values of normal variate (NV) with zero expectation and unit variance (ZEUV), where $F \in \mathbb{N}$. The assignment (2) gives the pixel noise SD on the k-th step of forming F matrices $\left\{\mathbf{A}_{\text{PDMI}}^{\langle k \rangle}\right\}_{k=1}^{F}$.

For accomplishing the training process we use the good-proven MATLAB training function "traingda" as one of the fastest training functions for TLP which is trained with backpropagation algorithm [33, 27]. In the PDMI training process, the input of TLP is fed with the training set

$$\left\{\tilde{\mathbf{P}}_i^{\langle \text{PDMI} \rangle}\right\}_{i=1}^{C+F} = \left\{\left\{\mathbf{A}\right\}_{l=1}^{C}, \left\{\mathbf{A}_{\text{PDMI}}^{\langle k \rangle}\right\}_{k=1}^{F}\right\} \tag{3}$$

of $C \in \mathbb{N}$ replicas of matrix $\mathbf{A}$ of non-distorted representatives and F matrices of PDMI by the set of identifiers (targets)

$$\left\{\mathbf{T}_i\right\}_{i=1}^{C+F} = \left\{\mathbf{I}\right\}_{i=1}^{C+F} \tag{4}$$

with identity $Q \times Q$ matrix $\mathbf{I}$. The set (3), being formed by (1) and (2), is passed through TLP with targets (4) for $J_{\text{pass}} \in \mathbb{N}$ times.

6.2 Model of STSMI

By the model of STSMI, the image successively is scaled, turned, and shifted. Such succession is necessary because it is running from the easy-to-model distortion type towards the worst distortion type.

A scaled image is formed by means of the MATLAB function "imresize". The map ρ, implementing this function partly, is applied to the image $\mathbf{A}_q$ as

$$\mathbf{A}_q^{\langle \text{S} \rangle}(k) = \rho\left(\mathbf{A}_q, \varsigma\left(\sigma_{\text{scale}}^{\langle k \rangle}\right)\right) \tag{5}$$

with the scale coefficient $\varsigma\left(\sigma_{\text{scale}}^{\langle k \rangle}\right)$ by SD

$$\sigma_{\text{scale}}^{\langle k \rangle} = \frac{k}{F} \cdot \sigma_{\text{scale}}^{\langle \max \rangle} \quad \forall\, k = \overline{1, F} \quad \text{for} \quad \sigma_{\text{scale}}^{\langle \max \rangle} > 0\,. \tag{6}$$

The scale coefficient

$$\varsigma\left(\sigma_{\text{scale}}^{\langle k \rangle}\right) = \sigma_{\text{scale}}^{\langle k \rangle} \xi_{\text{scale}}(k) + 1 \tag{7}$$

is determined by the value $\xi_{\text{scale}}(k)$ of NV with ZEUV raffled at the k-th stage of STSMI set formation. If occurs $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)\leqslant 0$ then the corresponding NV with ZEUV is re-raffled until $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)>0$. The input image $\mathbf{A}_q$ is enlarged by $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)$ times within the map (5) if $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)>1$; the input image $\mathbf{A}_q$ is reduced by $\dfrac{1}{\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)}$ times within the map (5) if $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)<1$; the input image $\mathbf{A}_q$ remains non-scaled if $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)=1$.

Another NV with ZEUV is raffled at the k-th stage of STSMI set formation for turning the scaled image $\mathbf{A}_q^{\langle \text{S}\rangle}(k)$. Its value $\xi_{\text{turn}}(k)$ is used within the MATLAB function "imrotate" implemented by the map τ. The procedure of turning the scaled image $\mathbf{A}_q^{\langle \text{S}\rangle}(k)$ of the q-th class is stated implicitly as

$$\mathbf{A}_q^{\langle \text{ST}\rangle}(k)=1-\tau\left(1-\mathbf{A}_q^{\langle \text{S}\rangle}(k),\,\beta\left(\sigma_{\text{turn}}^{\langle k\rangle}\right)\right), \tag{8}$$

where the map τ in (8) along with the argument $1-\mathbf{A}_q^{\langle \text{S}\rangle}(k)$ takes the turn angle $\beta\left(\sigma_{\text{turn}}^{\langle k\rangle}\right)$ by SD

$$\sigma_{\text{turn}}^{\langle k\rangle}=\frac{k}{F}\cdot\sigma_{\text{turn}}^{\langle \max\rangle}\quad\forall\ k=\overline{1,\,F}\quad\text{for}\quad\sigma_{\text{turn}}^{\langle \max\rangle}>0. \tag{9}$$

Actually, the map (8) takes the negative (pixel-inverted FMI) of the scaled image $\mathbf{A}_q^{\langle \text{S}\rangle}(k)$ and rotates the input image $1-\mathbf{A}_q^{\langle \text{S}\rangle}(k)$ by

$$\beta\left(\sigma_{\text{turn}}^{\langle k\rangle}\right)=\frac{180}{\pi}\sigma_{\text{turn}}^{\langle k\rangle}\xi_{\text{turn}}(k) \tag{10}$$

degrees around its center point. The image is turned in counterclockwise direction if $\beta\left(\sigma_{\text{turn}}^{\langle k\rangle}\right)>0$; for $\beta\left(\sigma_{\text{turn}}^{\langle k\rangle}\right)<0$ the image is turned clockwise; for $\beta\left(\sigma_{\text{turn}}^{\langle k\rangle}\right)=0$ the image remains unturned. The map τ in (8) returns the image which is inverted back.

Before shifting the scaled-turned image, it must have the source format $Y\times W$. By $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)\neq 1$, the scaled-turned image is the matrix $\mathbf{A}_q^{\langle \text{ST}\rangle}(k)$ of the intermediary format $V\times H$. The format of the matrices $\mathbf{A}_q^{\langle \text{ST}\rangle}(k)$ and $\mathbf{A}_q^{\langle \text{S}\rangle}(k)$ is the same. If $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)>1$ then the scaled-turned image is cropped. If cropped, then lines of their numbers

$$\left\{\left\{\overline{1,\,N_V}\right\},\left\{\overline{Y+1+N_V,\,V}\right\}\right\} \tag{11}$$

and columns of their numbers

$$\left\{\left\{\overline{1,\,N_H}\right\},\left\{\overline{W+1+N_H,\,H}\right\}\right\} \tag{12}$$

in the matrix $\mathbf{A}_q^{\langle \text{ST}\rangle}(k)$ are discarded. The integers

$$N_V=\eta\left(\frac{V-Y}{2}\right)+\left(\frac{1+\operatorname{sign}\zeta_V}{2}\cdot\operatorname{sign}\left|\zeta_V\right|\right)\cdot\operatorname{sign}\left[\frac{V}{2}-\eta\left(\frac{V}{2}\right)\right] \tag{13}$$

and

$$N_H=\eta\left(\frac{H-W}{2}\right)+\left(\frac{1+\operatorname{sign}\zeta_H}{2}\cdot\operatorname{sign}\left|\zeta_H\right|\right)\cdot\operatorname{sign}\left[\frac{H}{2}-\eta\left(\frac{H}{2}\right)\right] \tag{14}$$

for (11) and (12) are calculated by the values $\{\zeta_V,\,\zeta_H\}$ of two independent NV with ZEUV raffled every time when the function $\eta(x)$ returning the integer part of the number x is applied. If $\varsigma\left(\sigma_{\text{scale}}^{\langle k\rangle}\right)<1$ then $V<Y$ and $H<W$, and the scaled-turned image is contoured rectangularly with the background white color. For making this, the matrix $\mathbf{A}_q^{\langle \text{ST}\rangle}(k)$ is padded from the left for

$$N_{\text{left}} = \eta\left(\frac{W-H}{2}\right)+\left(\frac{1+\operatorname{sign}\zeta_H}{2}\cdot\operatorname{sign}\left|\zeta_H\right|\right)\cdot\operatorname{sign}\left[\frac{H}{2}-\eta\left(\frac{H}{2}\right)\right] \tag{15}$$

columns of ones and from the right for

$$N_{\text{right}} = W - H - N_{\text{left}} \tag{16}$$

columns of ones, and the matrix $\mathbf{A}_q^{\langle \text{ST}\rangle}(k)$ is padded from the top for

$$N_{\text{top}} = \eta\left(\frac{Y-V}{2}\right)+\left(\frac{1+\operatorname{sign}\zeta_V}{2}\cdot\operatorname{sign}\left|\zeta_V\right|\right)\cdot\operatorname{sign}\left[\frac{V}{2}-\eta\left(\frac{V}{2}\right)\right] \tag{17}$$

lines of ones and from the bottom for

$$N_{\text{bottom}} = Y - V - N_{\text{top}} \tag{18}$$

lines of ones.

The cropped or contoured scaled-turned image as $Y\times W$ matrix $\tilde{\mathbf{A}}_q^{\langle \text{ST}\rangle}(k)=\left[\tilde{a}_{uv}^{\langle q\rangle}(k)\right]_{Y\times W}$ is shifted horizontally and vertically. A shift constant consists of two components, horizontal and vertical, where SD

$$\sigma_{\text{shift}}^{\langle k\rangle} = \frac{k}{F}\cdot\sigma_{\text{shift}}^{\langle \max\rangle} \quad \forall\, k=\overline{1,\, F} \quad \text{and} \quad \sigma_{\text{shift}}^{\langle \max\rangle} > 0 \tag{19}$$

is used. For $Y\times W$ image, the horizontal pixel shift (HPS) is

$$s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right) = \eta\left(\gamma W\sigma_{\text{shift}}^{\langle k\rangle}\cdot\xi_{\text{hor}}(k)\right)\cdot\frac{1-\operatorname{sign}\left(\left|\eta\left(\gamma W\sigma_{\text{shift}}^{\langle k\rangle}\cdot\xi_{\text{hor}}(k)\right)\right|-W\right)}{2}+$$
$$+W\cdot\frac{1+\operatorname{sign}\left(\left|\eta\left(\gamma W\sigma_{\text{shift}}^{\langle k\rangle}\cdot\xi_{\text{hor}}(k)\right)\right|-W\right)}{2}, \tag{20}$$

where $\xi_{\text{hor}}(k)$ is a value of NV with ZEUV raffled at the k-th stage of STSMI set formation for HPS, and $\gamma>0$ is an HPS magnitude regulator. This regulator is used for vertical pixel shift (VPS) as well. Concurrently, VPS is

$$s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right) = \eta\left(\gamma Y\sigma_{\text{shift}}^{\langle k\rangle}\cdot\xi_{\text{ver}}(k)\right)\cdot\frac{1-\operatorname{sign}\left(\left|\eta\left(\gamma Y\sigma_{\text{shift}}^{\langle k\rangle}\cdot\xi_{\text{ver}}(k)\right)\right|-Y\right)}{2}+$$
$$+Y\cdot\frac{1+\operatorname{sign}\left(\left|\eta\left(\gamma Y\sigma_{\text{shift}}^{\langle k\rangle}\cdot\xi_{\text{ver}}(k)\right)\right|-Y\right)}{2}, \tag{21}$$

where $\xi_{\text{ver}}(k)$ is a value of NV with ZEUV raffled at the k-th stage of STSMI set formation for VPS. From previous experience, $\gamma\in\{0.1,\, 0.05\}$ or some about that for both (20) and (21).

Due to the horizontal shift, the matrix $\tilde{\mathbf{A}}_q^{\langle \text{ST}\rangle}(k)=\left[\tilde{a}_{uv}^{\langle q\rangle}(k)\right]_{Y\times W}$ is transformed into the matrix $\tilde{\mathbf{A}}_q^{\langle \text{ST-HPS}\rangle}(k)=\left[\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k)\right]_{Y\times W}$. For $s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)>0$ elements of the matrix $\tilde{\mathbf{A}}_q^{\langle \text{ST-HPS}\rangle}(k)$ are

$$\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k)=1 \quad \text{for} \quad u=\overline{1,\, Y} \quad \text{and} \quad v=\overline{1,\, s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)} \tag{22}$$

by

$$\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k)=\tilde{a}_{ut}^{\langle q\rangle}(k) \quad \text{at} \quad t=v-s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right) \quad \text{for} \quad u=\overline{1,\, Y} \quad \text{and} \quad v=\overline{s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)+1,\, W}. \tag{23}$$

For $s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)<0$ elements of the matrix $\tilde{\mathbf{A}}_q^{\langle \text{ST-HPS}\rangle}(k)$ are

$$\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k)=\tilde{a}_{ut}^{\langle q\rangle}(k) \quad \text{at} \quad t=v-s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right) \quad \text{for} \quad u=\overline{1,\, Y} \quad \text{and} \quad v=\overline{1,\, W+s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)} \tag{24}$$

by

$$\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k)=1 \quad \text{for} \quad u=\overline{1,\, Y} \quad \text{and} \quad v=\overline{W+s_{\text{hor}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)+1,\, W}. \tag{25}$$

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

$$\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k)=\tilde{a}_{uv}^{\langle q\rangle}(k) \quad \text{for} \quad u=\overline{1,Y} \quad \text{and} \quad v=\overline{1,W}. \tag{26}$$

After the horizontal shift, due to the vertical shift, the matrix $\tilde{\mathbf{A}}_q^{\langle \text{ST-HPS}\rangle}(k)$ is transformed into the matrix $\tilde{\mathbf{A}}_q^{\langle \text{STSMI}\rangle}(k)=\left[\tilde{a}_{uv}^{\langle q\text{-STSMI}\rangle}(k)\right]_{Y\times W}$. For $s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)>0$ elements of the matrix $\tilde{\mathbf{A}}_q^{\langle \text{STSMI}\rangle}(k)$ are

$$\tilde{a}_{uv}^{\langle q\text{-STSMI}\rangle}(k)=\tilde{a}_{rv}^{\langle q\text{-HPS}\rangle}(k) \quad \text{at} \quad r=u+s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right) \quad \text{for} \quad u=\overline{1,Y-s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)} \quad \text{and} \quad v=\overline{1,W} \tag{27}$$

by

$$\tilde{a}_{uv}^{\langle q\text{-STSMI}\rangle}(k)=1 \quad \text{for} \quad u=\overline{Y-s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)+1,Y} \quad \text{and} \quad v=\overline{1,W}. \tag{28}$$

For $s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)<0$ elements of the matrix $\tilde{\mathbf{A}}_q^{\langle \text{STSMI}\rangle}(k)$ are

$$\tilde{a}_{uv}^{\langle q\text{-STSMI}\rangle}(k)=1 \quad \text{for} \quad u=\overline{1,-s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)} \quad \text{and} \quad v=\overline{1,W} \tag{29}$$

by

$$\tilde{a}_{uv}^{\langle q\text{-STSMI}\rangle}(k)=\tilde{a}_{rv}^{\langle q\text{-HPS}\rangle}(k) \quad \text{at} \quad r=u+s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right) \quad \text{for} \quad u=\overline{-s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)+1,Y} \quad \text{and} \quad v=\overline{1,W}. \tag{30}$$

Clearly, for $s_{\text{ver}}\left(\sigma_{\text{shift}}^{\langle k\rangle}\right)=0$ the q-th horizontally shifted image is not shifted vertically:

$$\tilde{a}_{uv}^{\langle q\text{-STSMI}\rangle}(k)=\tilde{a}_{uv}^{\langle q\text{-HPS}\rangle}(k) \quad \text{for} \quad u=\overline{1,Y} \quad \text{and} \quad v=\overline{1,W}. \tag{31}$$

After every image became an STSMI, matrices of all STSMI are column-reshaped and concatenated into the matrix $\mathbf{A}_{\text{STSMI}}^{\langle k\rangle}=\left(\tilde{a}_{jq}^{\langle k\text{-STSMI}\rangle}\right)_{(Y\cdot W)\times Q}$, ready for including it into the training set or for adding the pixel noise distortion. In the STSMI training process, the input of TLP is fed with the training set

$$\left\{\tilde{\mathbf{P}}_i^{\langle \text{STSMI}\rangle}\right\}_{i=1}^{C+F}=\left(\left\{\mathbf{A}\right\}_{l=1}^{C},\left\{\mathbf{A}_{\text{STSMI}}^{\langle k\rangle}\right\}_{k=1}^{F}\right) \tag{32}$$

by targets (4), being passed through TLP for J_{pass} times.

With scaling by (5) — (7), turning by (8) — (10), cropping or contouring by (11) — (18), shifting by (19) — (31), the values of the corresponding SD in (6), (9), (19) are varied synchronously, like these SD are parts of a whole SD for forming randomized STSMI. This is realized by setting a ratio between ultimate values $\sigma_{\text{scale}}^{\langle \max\rangle}$, $\sigma_{\text{turn}}^{\langle \max\rangle}$, $\sigma_{\text{shift}}^{\langle \max\rangle}$. The ratio is maintained for the current values σ_{scale}, σ_{turn}, σ_{shift}.

6.3 Model of PDSTSMI

Model of PDSTSMI is in forming the matrix $\mathbf{A}_{\text{PDSTSMI}}^{\langle k\rangle}=\left(\tilde{a}_{jq}^{\langle k\text{-PDSTSMI}\rangle}\right)_{(Y\cdot W)\times Q}$ through the addition

$$\mathbf{A}_{\text{PDSTSMI}}^{\langle k\rangle}=\mathbf{A}_{\text{STSMI}}^{\langle k\rangle}+\sigma_{\text{pixel}}^{\langle k\rangle}\cdot\boldsymbol{\Xi} \tag{33}$$

by SD (2). And in the PDSTSMI training process, the input of TLP is fed with the training set

$$\left\{\tilde{\mathbf{P}}_i^{\langle \text{PDSTSMI}\rangle}\right\}_{i=1}^{C+F}=\left(\left\{\mathbf{A}\right\}_{l=1}^{C},\left\{\mathbf{A}_{\text{PDSTSMI}}^{\langle k\rangle}\right\}_{k=1}^{F}\right) \tag{34}$$

by targets (4), being passed through TLP for J_{pass} times.

The values of the corresponding SD in (2), (6), (9), (19) are varied synchronously, like these four SD are parts of a whole SD for forming randomized PDSTSMI. This is realized similarly to the STSMI model by setting a ratio between ultimate values $\sigma_{\text{pixel}}^{\langle \max\rangle}$, $\sigma_{\text{scale}}^{\langle \max\rangle}$, $\sigma_{\text{turn}}^{\langle \max\rangle}$, $\sigma_{\text{shift}}^{\langle \max\rangle}$. The ratio is maintained for the current values σ_{pixel}, σ_{scale}, σ_{turn}, σ_{shift}.

7 TLP Structure

TLP as mathematical object consists of two matrices and two vectors:

$$\left\{\left[w_{jl}\right]_{(Y\cdot W)\times H_{\mathrm{TLP}}}, \left[z_{lq}\right]_{H_{\mathrm{TLP}}\times Q}, \left[h_l\right]_{1\times H_{\mathrm{TLP}}}, \left[b_q\right]_{1\times Q}\right\} \tag{35}$$

by H_{TLP} neurons in TLP hidden layer. Matrices $\left[w_{jl}\right]_{(Y\cdot W)\times H_{\mathrm{TLP}}}$ and $\left[z_{lq}\right]_{H_{\mathrm{TLP}}\times Q}$ correspond to the input and hidden layer, respectively. Vectors $\left[h_l\right]_{1\times H_{\mathrm{TLP}}}$ and $\left[b_q\right]_{1\times Q}$ contain their biases. For object features $\left\{x_j\right\}_{j=1}^{Y\cdot W}$, the q-th output neuron value is

$$n_q = \xi\left(\sum_{l=1}^{H_{\mathrm{TLP}}}\xi\left(\sum_{j=1}^{Y\cdot W}x_j w_{jl} + h_l\right)z_{lq} + b_q\right) \quad \text{at} \quad q=\overline{1,\,Q} \tag{36}$$

by the logarithmic sigmoid transfer function (LSTF)

$$\xi(x) = \left(1+e^{-x}\right)^{-1}. \tag{37}$$

LSTF (37) is identically used in H_{TLP} hidden layer neurons and in Q output layer neurons. TLP classifier returns the class number q_* according to that

$$q_* \in \arg\max_{q=\overline{1,\,Q}}\left\{n_q\right\} \tag{38}$$

by (36).

For classifying FSTSO at bad distortions, it is sufficient to let $H_{\mathrm{TLP}}=250$ into the TLP structure. Besides, let $C=2$ and $F=8$ for the TLP training process, where two matrices and two vectors (35) are identified. The integer J_{pass} depends on types of distortions and their intensities. So, it shall be defined specifically.

8 Batch Testings of the Trained TLP

The trained TLP is tested on regular batch inputs. In this case, each class feeds the input of TLP for the same number of times. A batch testing is Q inputs, where each class is represented. These FSTSO feed the input of TLP through a range of a whole SD which defines intensities of distortions. To obtain statistically valid classification results, TLP shall be tested on no less than 100 batch testings. By B_{test} batch testings, CEP is calculated as

$$p_{\mathrm{error}} = \frac{100\sum_{t=1}^{B_{\mathrm{test}}}\sum_{q=1}^{Q}\operatorname{sign}\left|q-q_*\left(q_t\right)\right|}{Q\cdot B_{\mathrm{test}}} \tag{39}$$

by the classifier's response $q_*\left(q_t\right)$ to the q-th class representative in the t-th batch at the input of TLP. The response is determined as (38).

9 PDMI-Trained TLP for Classifying STSMI and PDSTSMI

Taken $\sigma_{\mathrm{pixel}}^{\langle\max\rangle}=1$, the integer $J_{\mathrm{pass}}=10$ is excellently practiced for training TLP with PDMI [38], which performs perfectly in classifying PDMI by the model (1) — (4). For seeing how PDMI-trained TLP classifies STSMI, let SD ratio be drawn from the equalities

$$\sigma_{\mathrm{shift}} = 5\sigma_{\mathrm{scale}} = 5\sigma_{\mathrm{turn}} \quad \text{at} \quad \sigma_{\mathrm{shift}}^{\langle\max\rangle}=0.5 \tag{40}$$

by $\gamma=0.1$ in (20) and (21). The equalities in (40) and $\gamma=0.1$ are assigned heuristically from previous experience. The maximum SD in assignment (40) has been taken carefully minimal.

For visualizing classification results over STSMI, there is the abscissa axis as shift-distortion SD σ_{shift} (Figure 2). We see in Figure 2 that CEP $p_{\mathrm{error}}\left(\sigma_{\mathrm{shift}}\right)$ over SD range $\left[0;\,0.5\right]$ increases intolerably. CEP over the segment

$[0; 0.125]$ is nonlinear. It resembles some shape of LSTF running to saturation over the segment $[0.075; 0.125]$. But starting off the point $\sigma_{\text{shift}} = 0.15$ rightwards, the CEP increasing is quasi-linear. Figure 2 shows the predictable fail of PDMI-trained TLP in classifying STSMI.

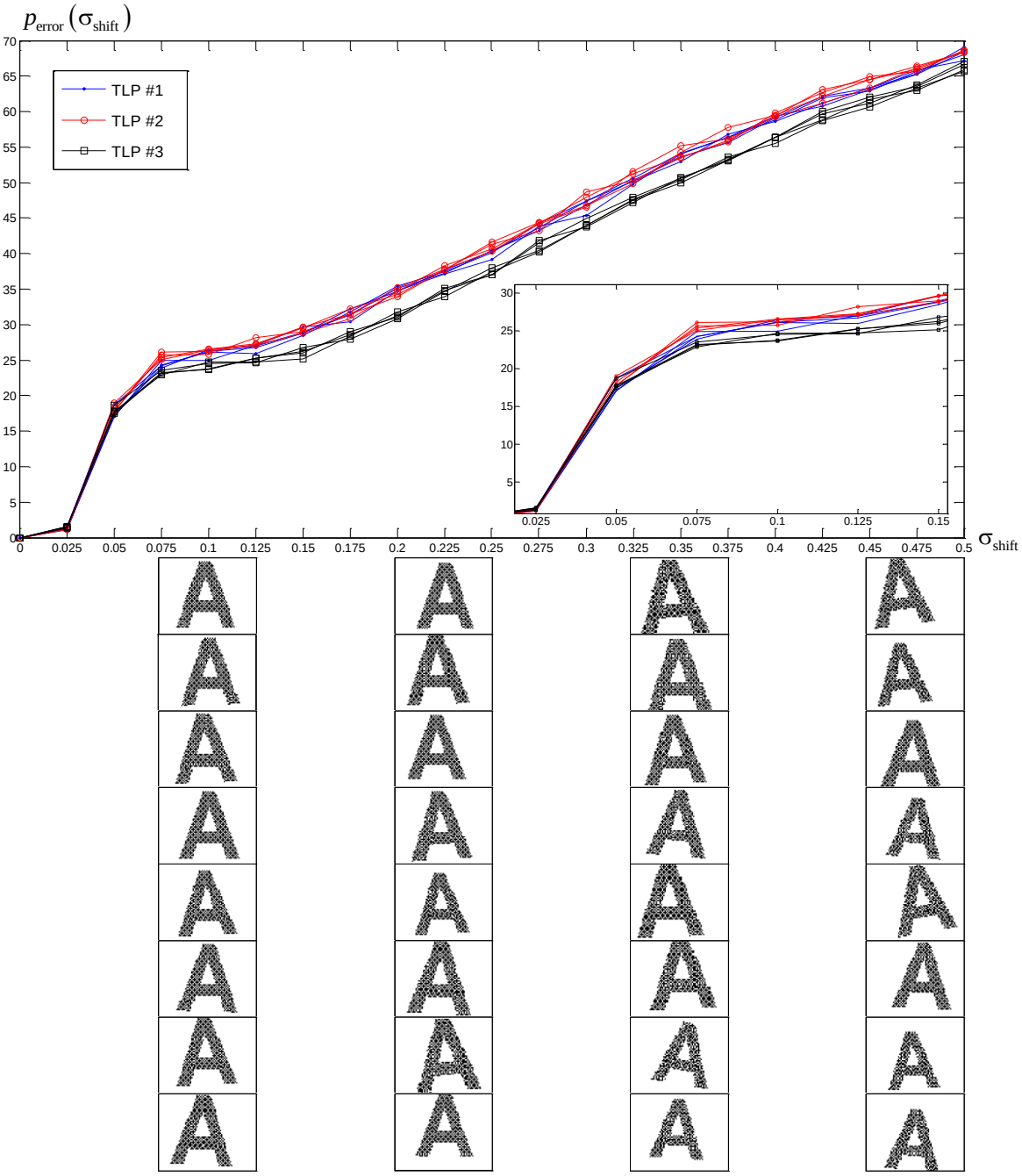

Figure 2: CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.5]$ by SD ratio drawn from (40) in STSMI after fourfold 200 batch testings of three PDMI-trained TLP; distortions of the letter "A" due to (40) are exemplified at $\sigma_{\text{shift}} \in \{0.125, 0.25, 0.375, 0.5\}$ along the abscissa axis

Obviously, it is expected that PDMI-trained TLP cannot classify PDSTSMI. For checking how PDMI-trained TLP classifies PDSTSMI, there is the heuristically assigned SD ratio drawn from the equalities

$$\sigma_{\text{shift}} = 5\sigma_{\text{scale}} = 5\sigma_{\text{turn}} = 0.5\sigma_{\text{pixel}} \quad \text{at} \quad \sigma_{\text{shift}}^{\langle \max \rangle} = 0.5 \tag{41}$$

by $\gamma = 0.1$ in (20) and (21). For visualizing classification results over PDSTSMI, the abscissa axis is as shift-distortion SD σ_{shift} (Figure 3). Figure 3 shows that the shape of $p_{\text{error}}(\sigma_{\text{shift}})$ for PDSTSMI is almost repeated after the shape of $p_{\text{error}}(\sigma_{\text{shift}})$ for STSMI, saying that neither STSMI nor PDSTSMI can be classified by PDMI-trained TLP.

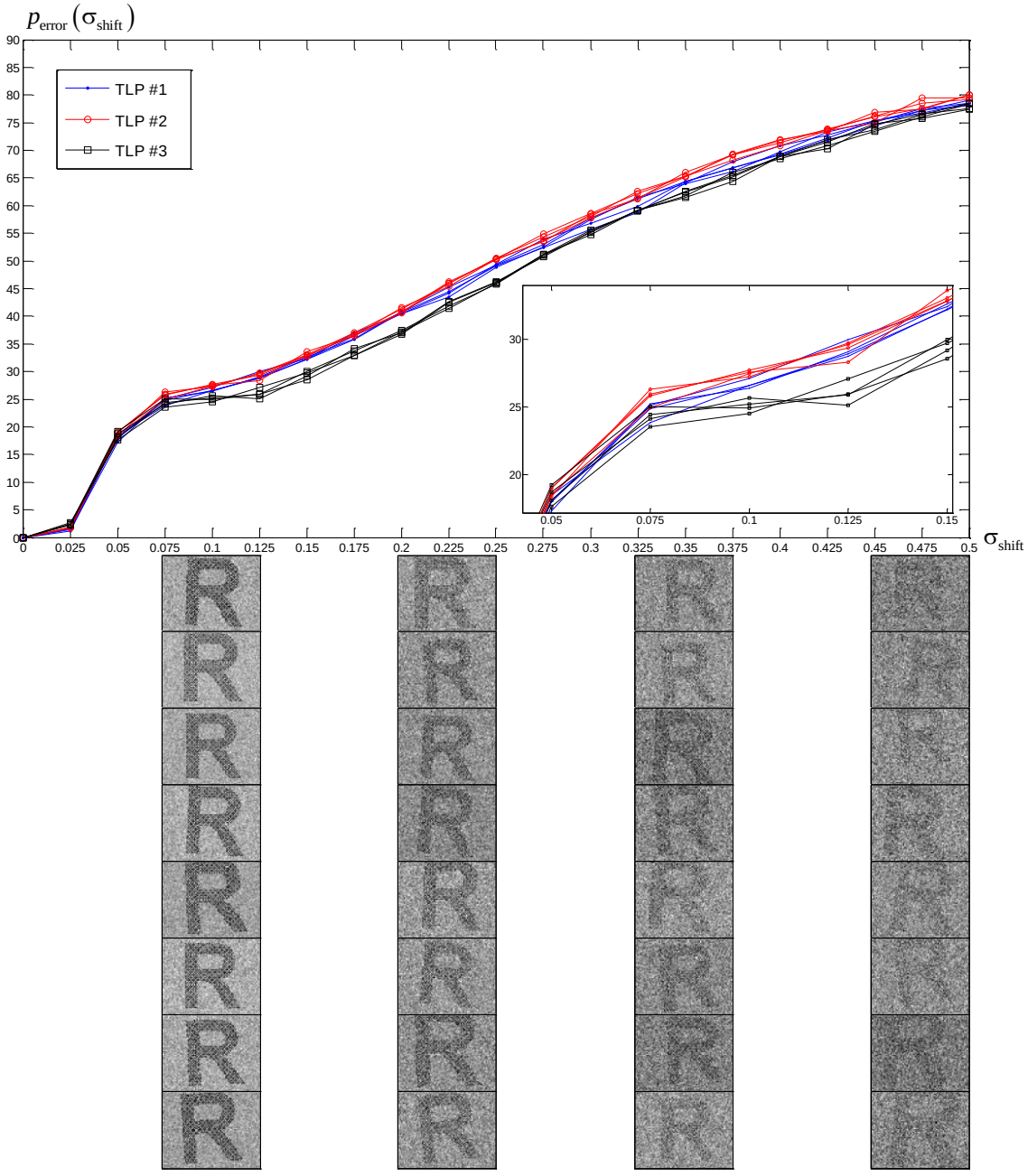

Figure 3: CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.5]$ by SD ratio drawn from (41) in PDSTSMI after fourfold 200 batch testings of three PDMI-trained TLP tested for Figure 2; distortions of the letter "R" due to (41) are exemplified at $\sigma_{\text{shift}} \in \{0.125, 0.25, 0.375, 0.5\}$ along the abscissa axis

Figures 2 and 3 prove that PDMI-trained TLP cannot classify STSMI or PDSTSMI, unless distortions are weak. The distortion weakness is when $\sigma_{\text{shift}} \leqslant 0.025$, what is very rare phenomenon. Hence, TLP is to be checked out

whether it can be trained with STSMI. Clearly, STSMI-trained TLP will be tested for classifying STSMI and PDSTSMI over the spoken above SD range $[0;\ 0.5]$ of the shift distortion. Classification of PDMI is of interest also.

10 STSMI-Trained TLP for Classifying STSMI

For training TLP with STSMI effectively, the integer J_{pass} for the training set (32) should be increased up from $J_{\text{pass}}=10$. The training set (32) is formed by (40). Polylines in Figure 4 show that STSMI-trained TLP classifies STSMI by (40) much better than PDMI-trained TLP does. CEP progressively decreases as J_{pass} increases, but the decrement progress early is fading out. By $\sigma_{\text{shift}}\leqslant 0.25$ or about that, STSMI-trained TLP classifies STSMI at high accuracy. Nevertheless, at higher SD, closer to $\sigma_{\text{shift}}=0.5$ point, STSMI-trained TLP classifies STSMI much worse. Besides, the STSMI training process is some overextended, having passes with unmet goals on reaching minimum gradient. Consequently, TLP cannot be trained with STSMI effectively, unless $\sigma_{\text{shift}}\leqslant 0.25$ or J_{pass} is increased enough and distortions are weak.

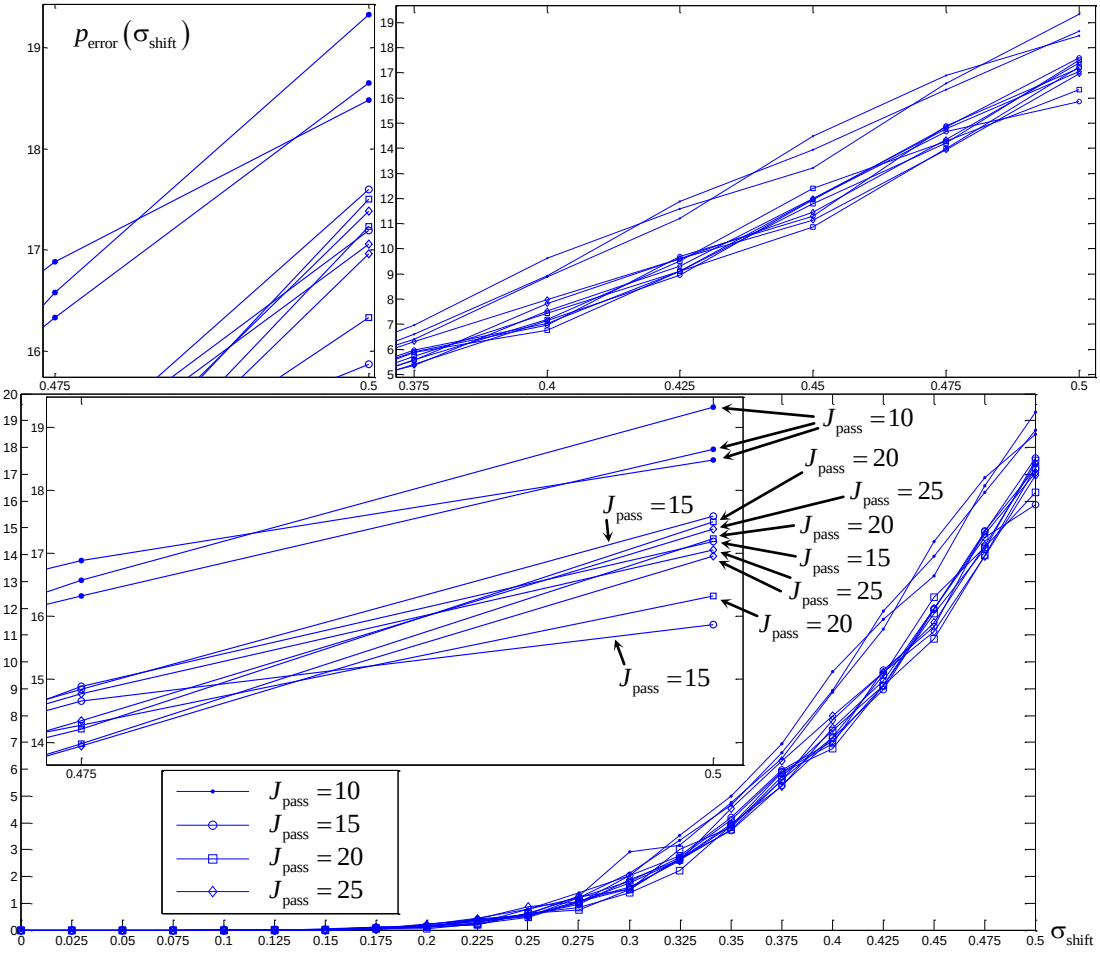

Figure 4: CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0;\ 0.5]$ by SD ratio drawn from (40) in STSMI after three-times-200-batch-tested the STSMI-trained TLP

After the obvious fail, STSMI-trained TLP is not reckoned to be a classifier of PDSTSMI (Figure 5). However, STSMI-trained TLP classifies PDMI quite good (Figure 6). But it's not a purpose of long STSMI-training.

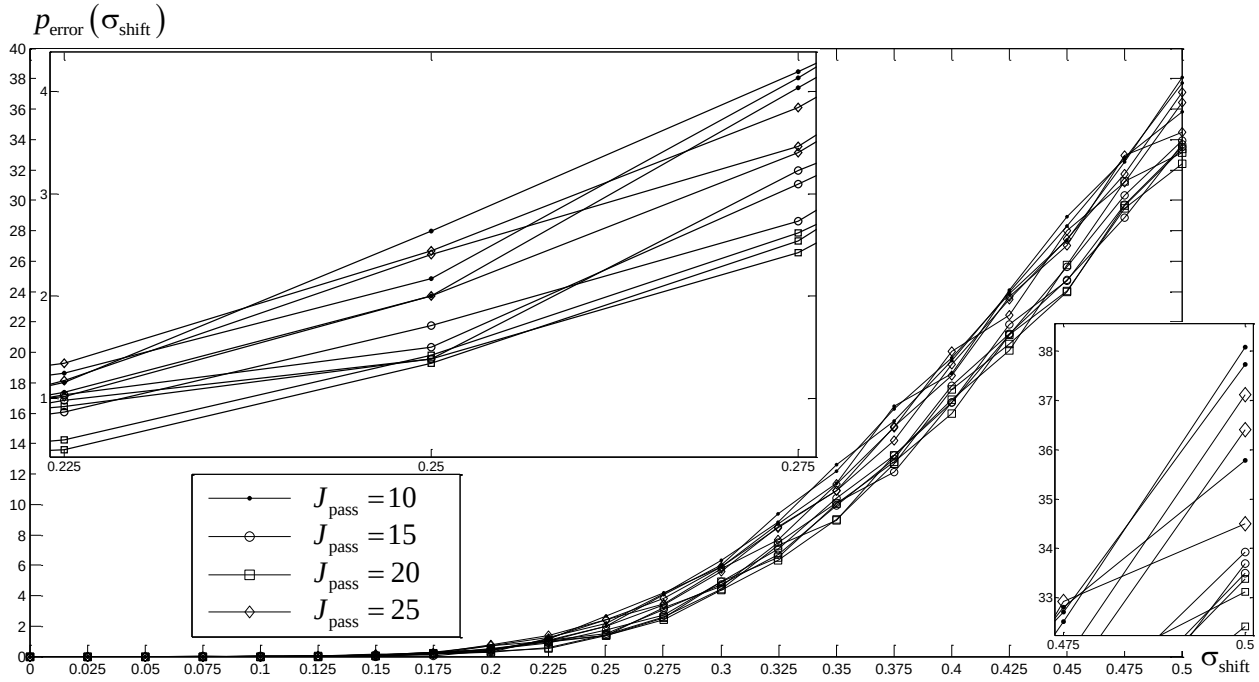

Figure 5: CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.5]$ by SD ratio drawn from (41) in PDSTSMI after three-times-200-batch-tested the STSMI-trained TLP

Now TLP is going to be trained with PDSTSMI by increasing the part of STSMI within the training set (34). That will be a try to use advantage of PDMI-trained TLP (in classifying PDMI) within PDSTSMI-trained TLP (for classifying STSMI).

11 PDSTSMI-Trained TLP for Classifying STSMI, PDSTSMI, and PDMI

Figure 4 and Figure 5 convince us of necessity for increasing the integer J_{pass} for the training set (34). Taken $J_{\text{pass}}=120$ on that necessity, the part of STSMI within the training set (34) is increased twice, starting with the equalities

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=0.125\sigma_{\text{pixel}} \quad \text{at} \quad \sigma_{\text{pixel}}^{\langle\max\rangle}=1. \tag{42}$$

By (42), the part of STSMI is just $7/47$ (the part of shift distortion is one eighth with respect to PDMI). Let the increment of $\sigma_{\text{shift}}^{\langle\max\rangle}$ be executed by

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=0.25\sigma_{\text{pixel}} \quad \text{at} \quad \sigma_{\text{pixel}}^{\langle\max\rangle}=1 \tag{43}$$

and

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=0.5\sigma_{\text{pixel}} \quad \text{at} \quad \sigma_{\text{pixel}}^{\langle\max\rangle}=1 \tag{44}$$

within the training set (34). Note, that the STSMI part increment is not twofold, as it might come in sight. By (43), the part of STSMI is $7/27$. And by (44), the part of STSMI is $7/17$. Further increment of the STSMI part lies in forming the training set (34) by

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=4\sigma_{\text{pixel}} \quad \text{at} \quad \sigma_{\text{pixel}}^{\langle\max\rangle}=0.125, \tag{45}$$

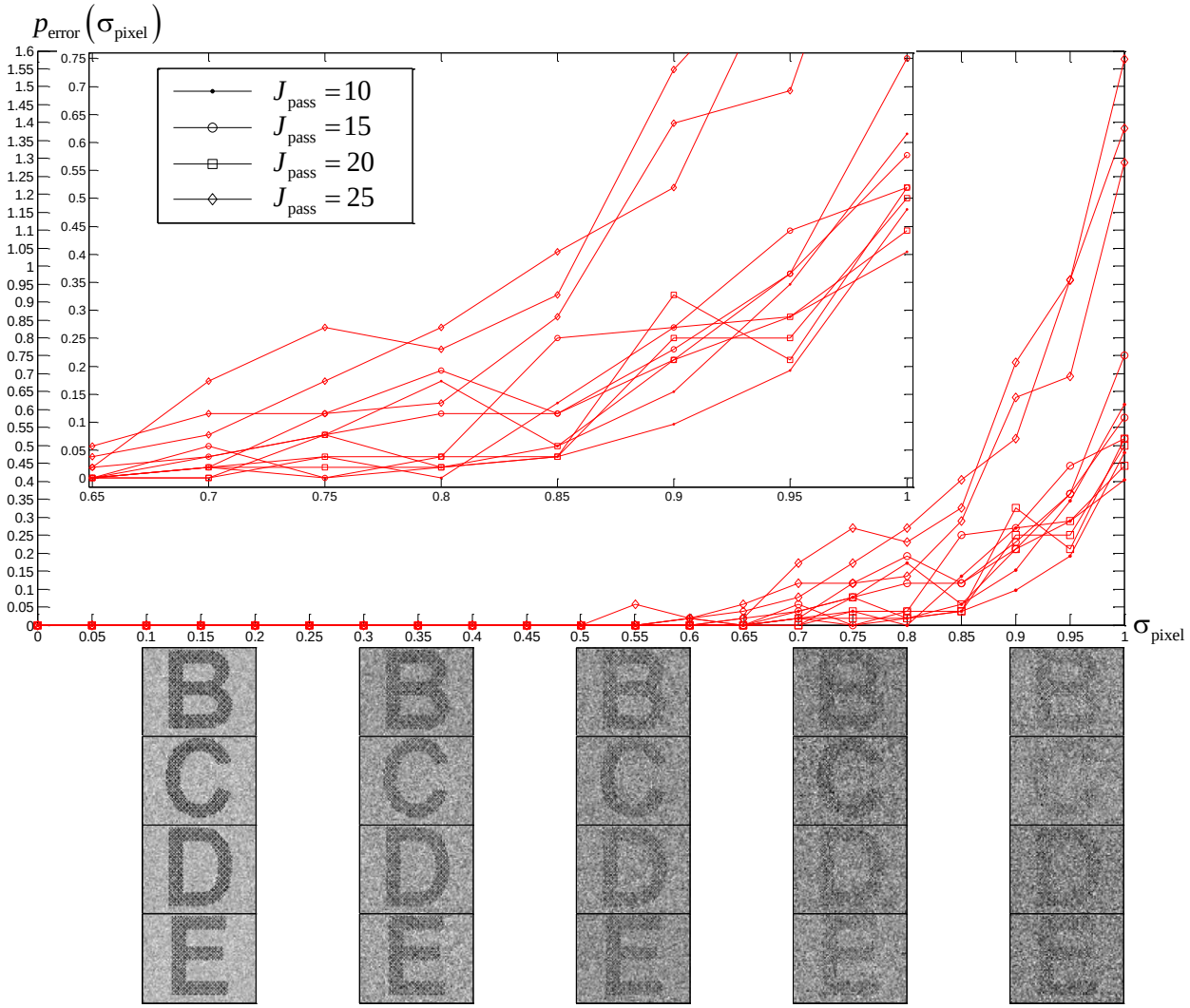

Figure 6: CEP $p_{\text{error}}(\sigma_{\text{pixel}})$ over SD range $[0;1]$ in PDMI after three-times-200-batch-tested the STSMI-trained TLP; PDMI are exemplified at $\sigma_{\text{pixel}}\in\{0.2,\ 0.4,\ 0.6,\ 0.8,\ 1\}$ along the abscissa axis

where the part of PDMI is just $5/33$ (rather than one fourth). Having tested the PDSTSMI-trained TLP with STSMI by (40), PDSTSMI training set configurations (42) — (44) are revealed to be poor, but the configuration

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=4\sigma_{\text{pixel}}\quad\text{at}\quad\sigma_{\text{pixel}}^{\langle\max\rangle}=0.25 \tag{46}$$

proves to be even better than the configuration (45), not changing the PDMI part (Figure 7).

Figure 7 hints and it is experienced the ratio

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=4\sigma_{\text{pixel}} \tag{47}$$

at some $\sigma_{\text{pixel}}^{\langle\max\rangle}>0.125$ is the closely-best for classifying PDSTSMI. Thereafter, 33 TLP have been PDSTSMI-trained at

$$\sigma_{\text{pixel}}^{\langle\max\rangle}\in\{0.125,\ 0.1875,\ 0.25\} \tag{48}$$

and various J_{pass} (Table 1) and tested (Figure 8). Then they are tested by

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}\quad\text{at}\quad\sigma_{\text{shift}}^{\langle\max\rangle}=0.75\,, \tag{49}$$

and only 14 TLP among these 33 classifiers perform accurately at $p_{\text{error}}(0.5) < 4$ (Figure 9), although CEP by $\sigma_{\text{shift}} > 0.5$ is linearly increasing.

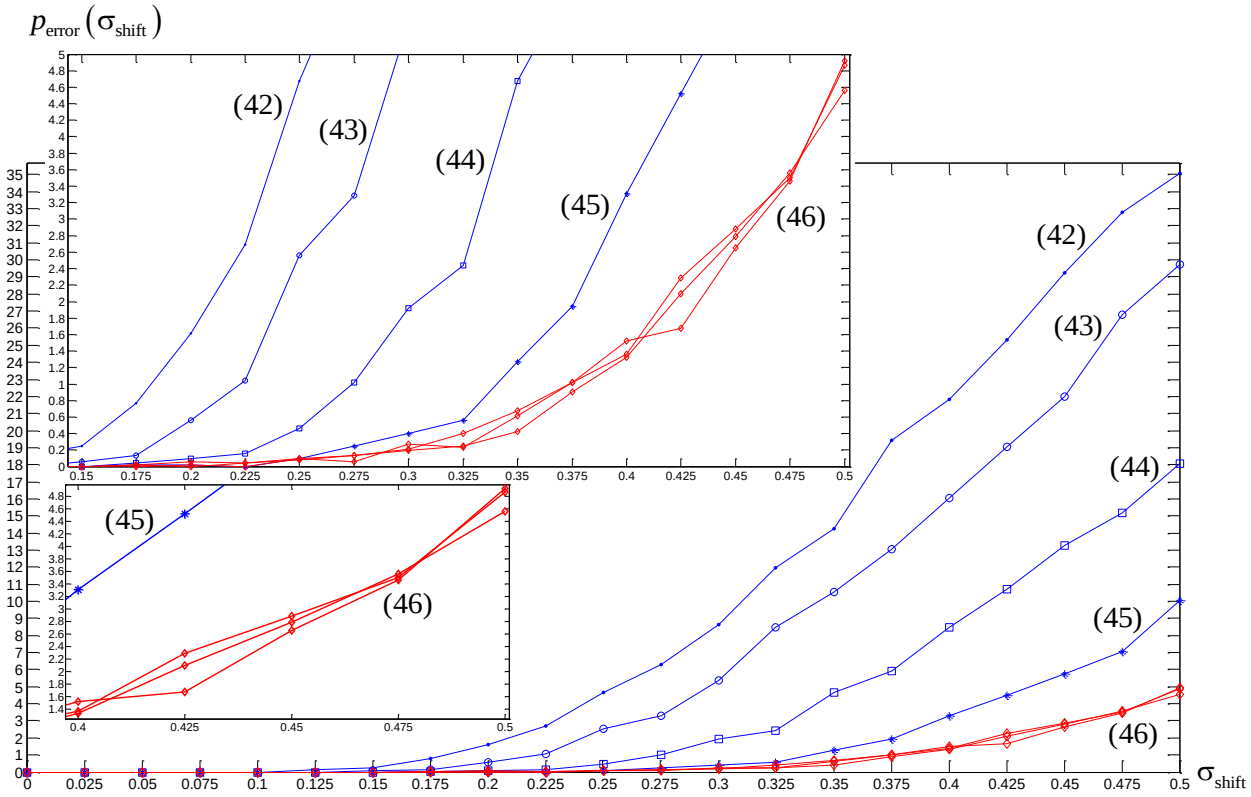

Figure 7: Gradual decrement of CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.5]$ by SD ratio drawn from (40) in STSMI, where configurations (42) — (46) of PDSTSMI training set (34) are applied (TLP with the best-configured PDSTSMI training set has been tested triply)

Table 1: Configurations of PDSTSMI training set (34) in trying to identify a high-accurate TLP classifier over STSMI

Number of TLP	$\sigma_{\text{pixel}}^{\langle\max\rangle}$	$\sigma_{\text{shift}}^{\langle\max\rangle}$	J_{pass}	Number of TLP	$\sigma_{\text{pixel}}^{\langle\max\rangle}$	$\sigma_{\text{shift}}^{\langle\max\rangle}$	J_{pass}	Number of TLP	$\sigma_{\text{pixel}}^{\langle\max\rangle}$	$\sigma_{\text{shift}}^{\langle\max\rangle}$	J_{pass}
1	0.1875	0.75	280	12	0.1875	0.75	250	23	0.125	0.5	260
2	0.125	0.5	220	13	0.1875	0.75	300	24	0.25	1.0	120
3	0.25	1.0	210	14	0.1875	0.75	300	25	0.25	1.0	140
4	0.1875	0.75	100	15	0.1875	0.75	350	26	0.25	1.0	160
5	0.1875	0.75	150	16	0.1875	0.75	350	27	0.25	1.0	170
6	0.1875	0.75	200	17	0.125	0.5	210	28	0.25	1.0	180
7	0.1875	0.75	200	18	0.125	0.5	210	29	0.25	1.0	190
8	0.1875	0.75	210	19	0.125	0.5	220	30	0.25	1.0	200
9	0.1875	0.75	230	20	0.125	0.5	230	31	0.25	1.0	210
10	0.1875	0.75	230	21	0.125	0.5	240	32	0.25	1.0	220
11	0.1875	0.75	250	22	0.125	0.5	250	33	0.25	1.0	230

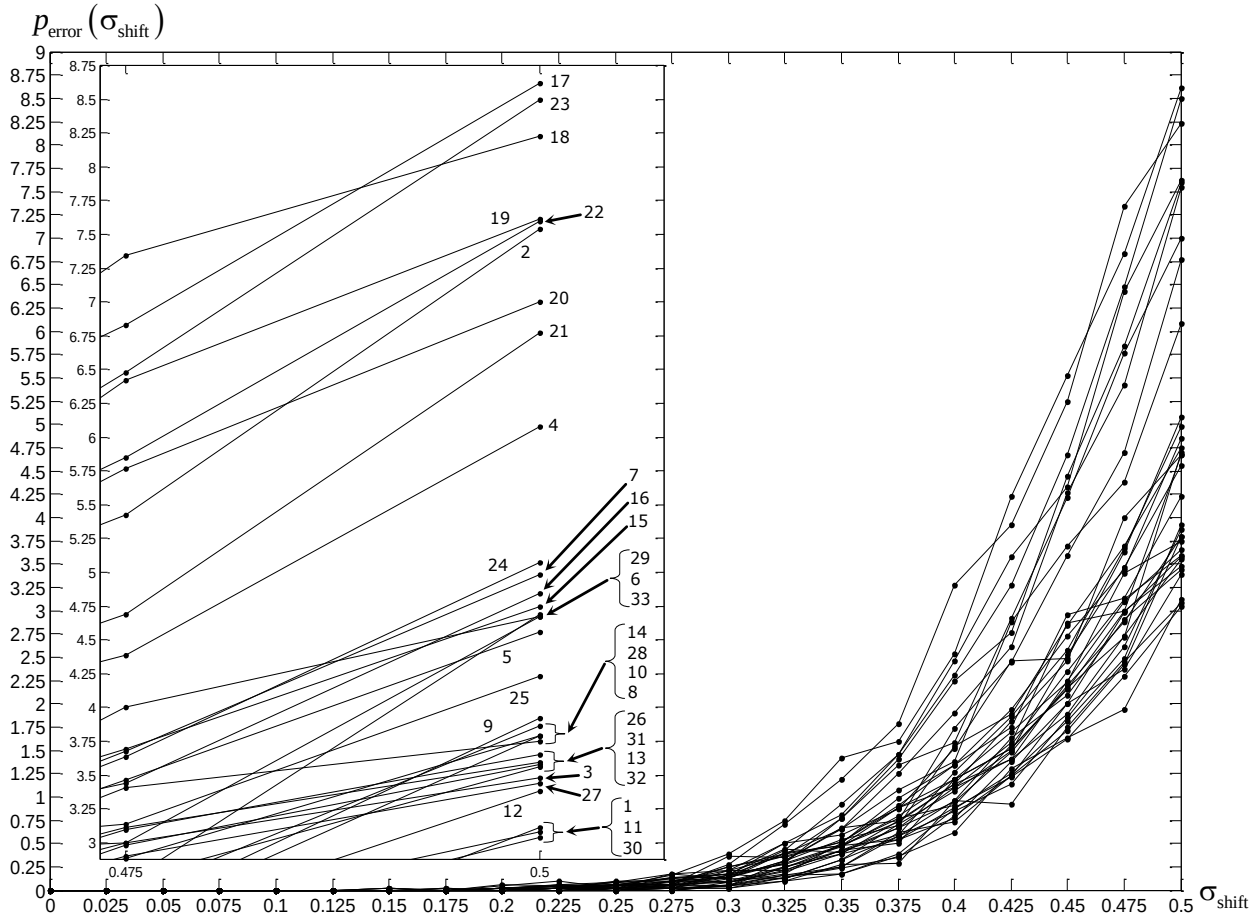

Figure 8: Scattering of CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.5]$ by SD ratio drawn from (40) in STSMI, where each of 33 PDSTSMI-trained TLP (numbered in Table 1) has been 200-batch-tested

Comparison of Figure 9 and Figure 4 reports that PDSTSMI-trained TLP performance excels STSMI-trained TLP performance. The performance is improved if $\sigma_{\text{shift}}^{\langle\max\rangle}$ is increased: 15 TLP in Figure 9 have been PDSTSMI-trained at $\sigma_{\text{shift}}^{\langle\max\rangle}=0.75$ and $\sigma_{\text{shift}}^{\langle\max\rangle}=1$, while all the worst TLP in Figure 8 (except TLP #4, most likely, because of 100 passes) have been PDSTSMI-trained at $\sigma_{\text{shift}}^{\langle\max\rangle}=0.5$. But increasing both $\sigma_{\text{shift}}^{\langle\max\rangle}$ and $\sigma_{\text{pixel}}^{\langle\max\rangle}$ impairs capability to classify PDSTSMI by

$$\sigma_{\text{shift}}=5\sigma_{\text{scale}}=5\sigma_{\text{turn}}=0.5\sigma_{\text{pixel}} \quad \text{at} \quad \sigma_{\text{shift}}^{\langle\max\rangle}=0.75 \tag{50}$$

and, especially, PDMI (Figure 10 and Figure 11, respectively).

In particular, all the worst TLP in Figure 10 have been PDSTSMI-trained at $\sigma_{\text{shift}}^{\langle\max\rangle}=1$. And each of 15 PDSTSMI-trained TLP in Figure 11 classifies PDMI. This implies that, for classifying STSMI, TLP should be trained at greater $\sigma_{\text{shift}}^{\langle\max\rangle}$ and $\sigma_{\text{pixel}}^{\langle\max\rangle}$ holding their ratio about (47). On the other hand, if PDSTSMI or PDMI are classified, TLP should be trained at lower $\sigma_{\text{shift}}^{\langle\max\rangle}$.

Consequently, if STSMI solely are expected, one of TLP #3, #27, #30, #31 numbered after Table 1 is suitable. Figures 8 and 9 show that these TLP perform at $p_{\text{error}}(0.5)<3.75$ and $p_{\text{error}}(0.75)<19.5$. Otherwise, if PDMI distortions are unavoidable, then a compromise version is TLP #21, whose performance is better over PDSTSMI (Figure 12) and PDMI (Figure 13). This classifier is coarser over STSMI, though (Figure 12).

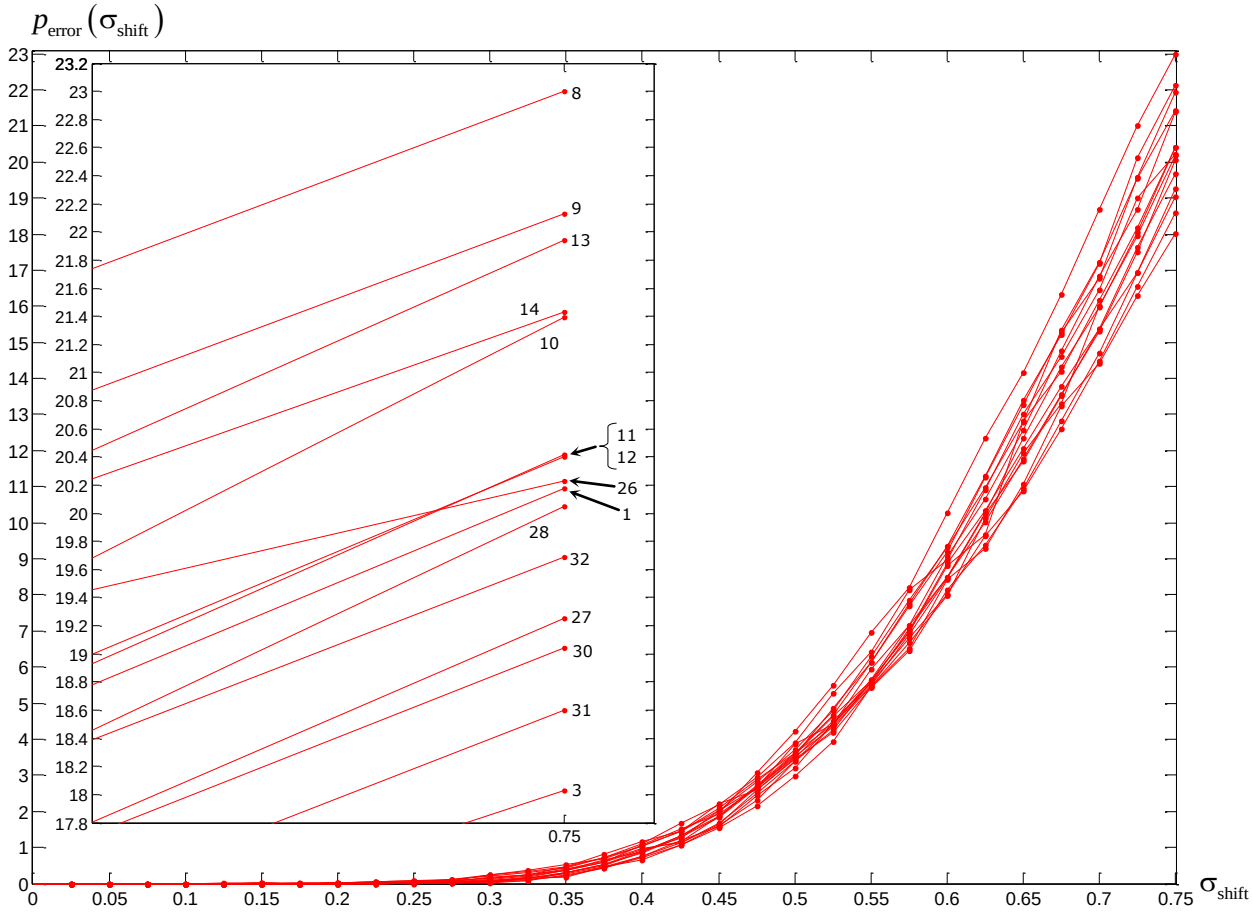

Figure 9: CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.75]$ by SD ratio drawn from (49) in STSMI, where each of 15 PDSTSMI-trained TLP (numbered in Table 1 and performing at lower CEP due to Figure 8) has been 800-batch-tested

Eventually, operation speed of PDSTSMI-trained TLP is alright since it takes about 48 seconds to classify 10000 60×80 FSTSO on Intel® Core™ i3-4150 CPU @ 3.50GHz by 4 GB of RAM (64-bit Windows 7). When parallelizing on two CPU cores, it takes 29 seconds. DLC and, particularly, neocognitron, on the similar system perform slower [37, 26]. Though parallelization helps in acceleration of DLC, its cost and resources consumption compared to TLP grow considerably.

12 Conclusion

Selection of a classifier over DDO addresses to coalescence of accuracy, rapidity, and simplicity. If FSTSO emulating DDO are 60×80 STSMI, then using PDSTSMI-trained TLP #3, #27, #30, #31 is effective at any intensity of distortions. If types of distortions are uncertain, the compromised TLP #21 is generally applicable. However, if the PDMI part is slight, then PDSTSMI-trained TLP #3, #27, #30, #31 are suitable.

If DDO are FSTSO whose TNOF is about 4800 and classes' number is about 26, then those listed TLP substitute DLC as well. And by a-few-percent-deviation from 4800 and 26, CEP behavior (Figures 9, 10, 11, 12, 13) mustn't change excessively. This means TLP which is PDSTSMI-trained at $\sigma_{\text{shift}}^{\langle\max\rangle}=1$ by (47) is capable to substitute DLC in classifying FSTSO, performing with 96 % accuracy at medium intensity of distortions (at $\sigma_{\text{shift}}=0.5$ and the respective magnitudes of related sigmas).

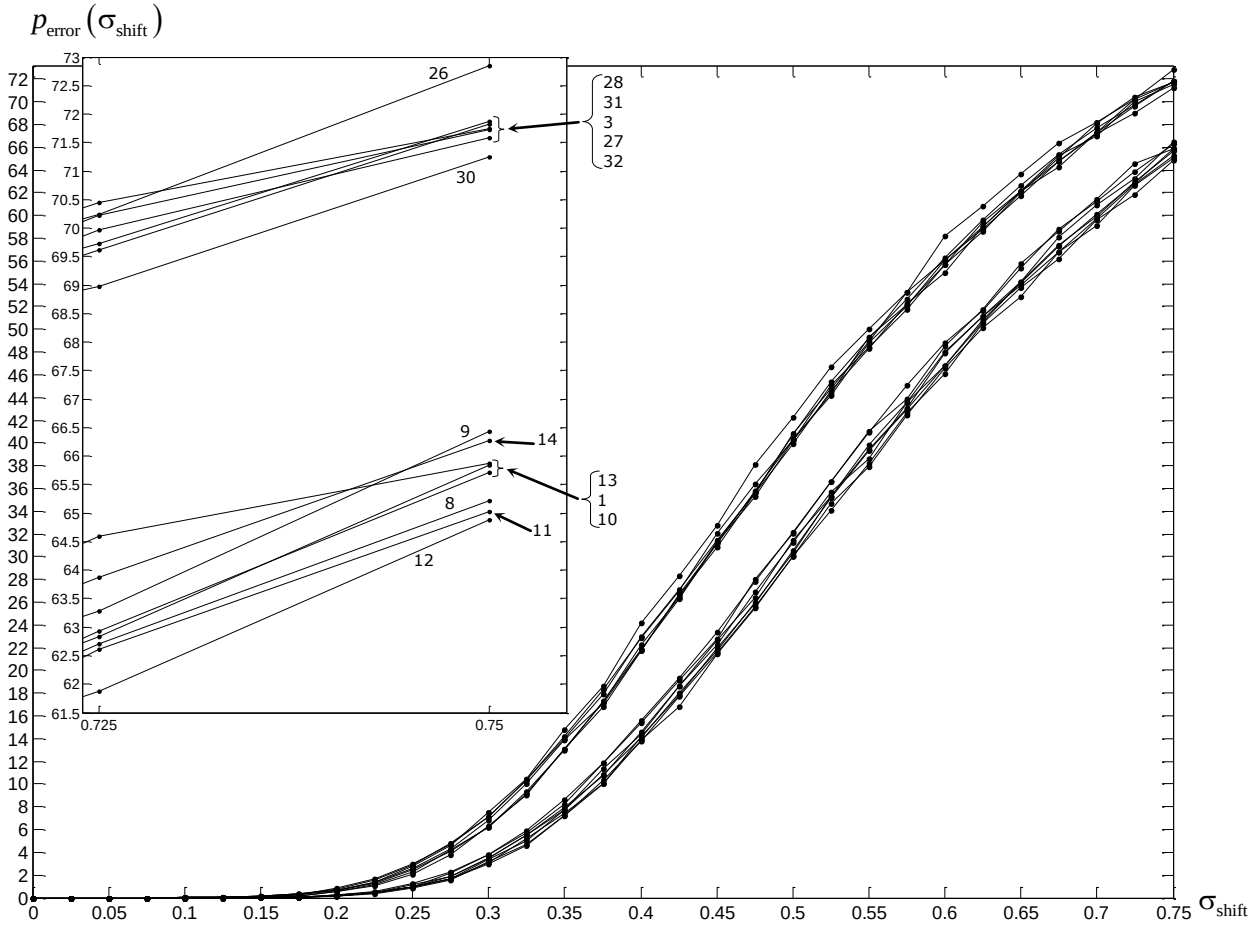

Figure 10: CEP $p_{\text{error}}(\sigma_{\text{shift}})$ over SD range $[0; 0.75]$ by SD ratio drawn from (50) in PDSTSMI, where each of 15 PDSTSMI-trained TLP in Figure 9 has been 800-batch-tested

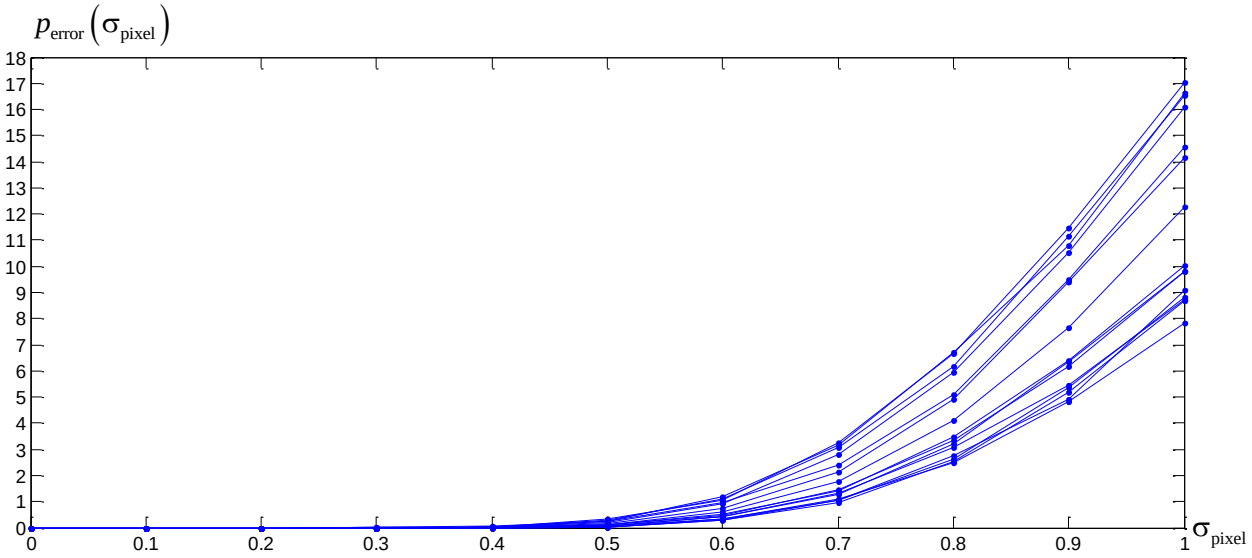

Figure 11: Unsatisfactory CEP $p_{\text{error}}(\sigma_{\text{pixel}})$ over SD range $[0; 1]$ in PDMI, where each of 15 PDSTSMI-trained TLP in Figures 9 and 10 has been 800-batch-tested

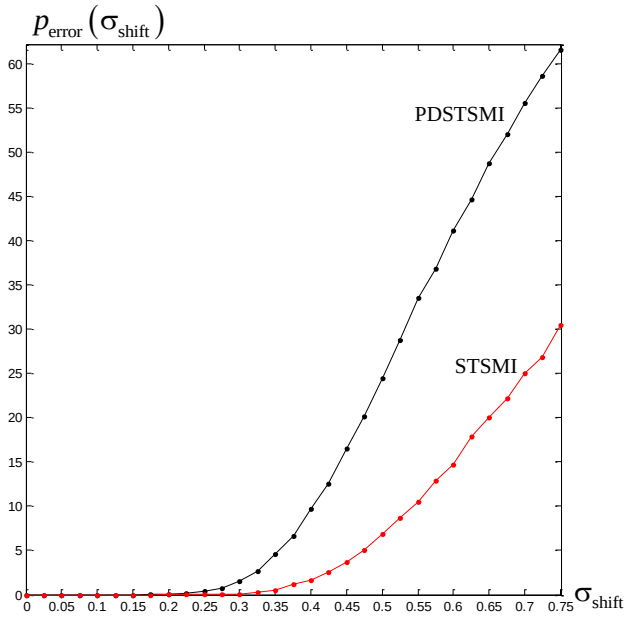

Figure 12: Performance of TLP #21 (800-batch-tested) over SD ranges $[0; 0.75]$ in STSMI and PDSTSMI by their SD ratios drawn from (49) and (50), respectively

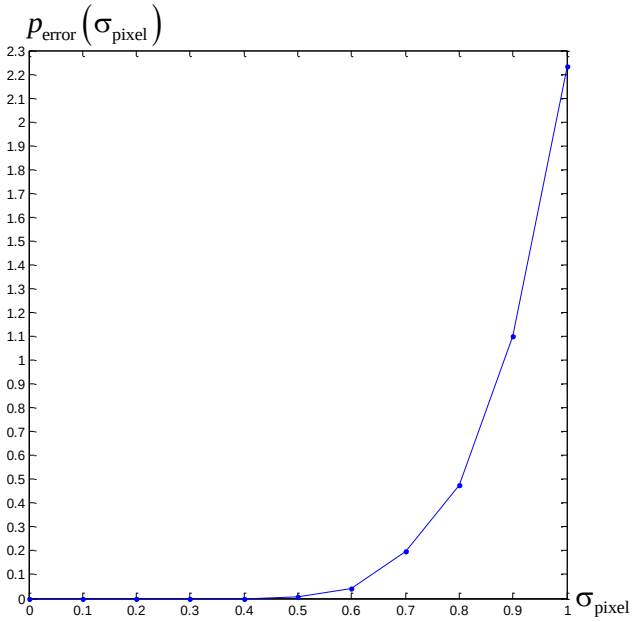

Figure 13: Performance of TLP #21 (800-batch-tested) over SD range $[0; 1]$ in PDMI; the performance is three times better than the best performance in Figure 11

Even if deviation from TNOF 4800 and $Q = 26$ is bigger, TLP should be trained with DDO by additional feature distortions (AFD), the specific embedding into the TLP training process. AFD, implying the addition (33), regularize the TLP training process in order to make TLP classify DDO more accurately like it classifies PDMI. This, however, drags the training process longer. For non-flat objects, say, 3D objects (data) or multidimensional data, the matrix Ξ is flat anyway whose elements are values of NV with ZEUV. And the matrix Ξ format is always $L \times Q$ by TNOF L.

Those good TLP for classifying FSTSO by AFD in training (#3, #21, #27, #30, #31) were obtained under SD ratio (47), although this ratio is scarcely the best. CEP (39) depends both on distortion intensities and ratios between them. Moreover, $\sigma_{\text{pixel}}^{\langle \max \rangle}$ for the training set influences on CEP, so choosing $\sigma_{\text{pixel}}^{\langle \max \rangle}$ mustn't be necessarily according to (48). Therefore, the ratio of sigmas could be improved by the criterion of CEP minimization. And this is a way to identify a TLP classifier whose CEP is lower than $p_{\text{error}}(\sigma_{\text{shift}})$ in Figure 9.

To advance the work in that way, a problem with three coefficients at sigmas

$$\left\{\sigma_{\text{shift}}, \sigma_{\text{scale}}, \sigma_{\text{turn}}, \sigma_{\text{pixel}}\right\} \tag{51}$$

in positive octant of three-dimensional Euclidean space is to be solved. The fourth coefficient is 1 at one of the sigmas (51). The advanced PDSTSMI-trained TLP would be more effective over DDO. Further to effectiveness of such TLP, an advanced TLP-based DLC could be configured, whose layers might be actually TLP optimized and trained by AFD.

Acknowledgments

This work was technically supported by the Center of parallel computations at Khmelnitskiy National University, Ukraine.

References

[1] Anastassiou, G.A., Multivariate sigmoidal neural network approximation, *Neural Networks*, vol.24, no.4, pp.378–386, 2011.

[2] Arulampalam, G., and A. Bouzerdoum, A generalized feedforward neural network architecture for classification and regression, *Neural Networks*, vol.16, nos.5–6, pp.561–568, 2003.

[3] Asadi, S., Hadavandi, E., Mehmanpazir, F., and M.M. Nakhostin, Hybridization of evolutionary Levenberg—Marquardt neural networks and data pre-processing for stock market prediction, *Knowledge-Based Systems*, vol.35, pp.245–258, 2012.

[4] Bai, J., Wu, Y., Zhang, J., and F. Chen, Subset based deep learning for RGB-D object recognition, *Neurocomputing*, vol.165, pp.280–292, 2015.

[5] Bordignon, F., and F. Gomide, Uninorm based evolving neural networks and approximation capabilities, *Neurocomputing*, vol.127, pp.13–20, 2014.

[6] Cao, F., Lin, S., and Z. Xu, Approximation capability of interpolation neural networks, *Neurocomputing*, vol.74, nos.1–3, pp.457–460, 2010.

[7] Castillo, P.A., Merelo, J.J., Arenas, M.G., and G. Romero, Comparing evolutionary hybrid systems for design and optimization of multilayer perceptron structure along training parameters, *Information Sciences*, vol.177, no.14, pp.2884–2905, 2007.

[8] Chairez, I., Finite time convergent learning law for continuous neural networks, *Neural Networks*, vol.50, pp.175–182, 2014.

[9] Chen, Z., Cao, F., and J. Hu, Approximation by network operators with logistic activation functions, *Applied Mathematics and Computation*, vol.256, pp.565–571, 2015.

[10] Cireşan, D., Meier, U., Masci, J., and J. Schmidhuber, Multi-column deep neural network for traffic sign classification, *Neural Networks*, vol.32, pp.333–338, 2012.

[11] Costarelli, D., and R. Spigler, Approximation results for neural network operators activated by sigmoidal functions, *Neural Networks*, vol.44, pp.101–106, 2013.

[12] Costarelli, D., and R. Spigler, Convergence of a family of neural network operators of the Kantorovich type, *Journal of Approximation Theory*, vol.185, pp.80–90, 2014.

[13] Fan, J., Zhang, J., Mei, K., Peng, J., and L. Gao, Cost-sensitive learning of hierarchical tree classifiers for large-scale image classification and novel category detection, *Pattern Recognition*, vol.48, no.5, pp.1673–1687, 2015.

[14] Fukushima, K., 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.

[15] Fukushima, K., Neural network model for selective attention in visual pattern recognition and associative recall, *Applied Optics*, vol.26, no.23, pp.4985–4992, 1987.

[16] Fukushima, K., Neocognitron: a hierarchical neural network capable of visual pattern recognition, *Neural Networks*, vol.1, no.2, pp.119–130, 1988.

[17] Fukushima, K., Neocognitron for handwritten digit recognition, *Neurocomputing*, vol.51, pp.161–180, 2003.

[18] Fukushima, K., Increasing robustness against background noise: visual pattern recognition by a neocognitron, *Neural Networks*, vol.24, no.7, pp.767–778, 2011.

[19] Fukushima, K., Artificial vision by multi-layered neural networks: neocognitron and its advances, *Neural Networks*, vol.37, pp.103–119, 2013.

[20] Fukushima, K., Training multi-layered neural network neocognitron, *Neural Networks*, vol.40, pp.18–31, 2013.

[21] Hagan, M.T., and M.B. Menhaj, Training feedforward networks with the Marquardt algorithm, *IEEE Transactions on Neural Networks*, vol.5, no.6, pp.989–993, 1994.

[22] Hagiwara, K., Hayasaka, T., Toda, N., Usui, S., and K. Kuno, Upper bound of the expected training error of neural network regression for a Gaussian noise sequence, *Neural Networks*, vol.14, no.10, pp.1419–1429, 2001.

[23] Haykin, S., *Neural Networks: A Comprehensive Foundation*, Prentice Hall, New Jersey, 1999.

[24] Huang, J., Li, Y.-F., and M. Xie, An empirical analysis of data preprocessing for machine learning-based software cost estimation, *Information and Software Technology*, vol.67, pp.108–127, 2015.

[25] Ismailov, V.E., On the approximation by neural networks with bounded number of neurons in hidden layers, *Journal of Mathematical Analysis and Applications*, vol.417, no.2, pp.963–969, 2014.

[26] Kangin, D., Kolev, G., and A. Vikhoreva, Further parameters estimation of neocognitron neural network modification with FFT convolution, *Journal of Telecommunication, Electronic and Computer Engineering*, vol.4, no.2, pp.21–26, 2012.

[27] Kathirvalavakumar, T., and S. Jeyaseeli Subavathi, Neighborhood based modified backpropagation algorithm using adaptive learning parameters for training feedforward neural networks, *Neurocomputing*, vol.72, nos.16–18, pp.3915–3921, 2009.

[28] Kim, S., Choi, Y., and M. Lee, Deep learning with support vector data description, *Neurocomputing*, vol.165, pp.111–117, 2015.

[29] Lillo-Castellano, J.M., Mora-Jiménez, I., Figuera-Pozuelo, C., and J.L. Rojo-Álvarez, Traffic sign segmentation and classification using statistical learning methods, *Neurocomputing*, vol.153, pp.286–299, 2015.

[30] Lin, S., Rong, Y., and Z. Xu, Multivariate Jackson-type inequality for a new type neural network approximation, *Applied Mathematical Modelling*, vol.38, no.24, pp.6031–6037, 2014.

[31] Ma, J., Zheng, L., Dong, M., He, X., Guo, M., Yaguchi, Y., and R. Oka, A segmentation-free method for image classification based on pixel-wise matching, *Journal of Computer and System Sciences*, vol.79, no.2, pp.256–268, 2013.

[32] Mei, K., Dong, P., Lei, H., and J. Fan, A distributed approach for large-scale classifier training and image classification, *Neurocomputing*, vol.144, pp.304–317, 2014.

[33] Moller, M.F., A scaled conjugate gradient algorithm for fast supervised learning, *Neural Networks*, vol.6, no.4, pp.525–533, 1993.

[34] Mrazova, I., and M. Kukacka, Can deep neural networks discover meaningful pattern features?, *Procedia Computer Science*, vol.2, pp.194–199, 2012.

[35] Nawi, N.M., Atomi, W.H., and M.Z. Rehman, The effect of data pre-processing on optimized training of artificial neural networks, *Procedia Technology*, vol.11, pp.32–39, 2013.

[36] Plaza, J., Plaza, A., Perez, R., and P. Martinez, On the use of small training sets for neural network-based characterization of mixed pixels in remotely sensed hyperspectral images, *Pattern Recognition*, vol.42, no.11, pp.3032–3045, 2009.

[37] Poli, G., and J.H. Saito, Parallel face recognition processing using neocognitron neural network and GPU with CUDA high performance architecture, *Face Recognition*, edited by Milos Oravec, InTech, Rijeka, Croatia, pp.381–404, 2010.

[38] Romanuke, V.V., 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, pp.38–54, 2013.

[39] Schmidhuber, J., Deep learning in neural networks: an overview, *Neural Networks*, vol.61, pp.85–117, 2015.

[40] Shao, H., Wang, J., Liu, L., Xu, D., and W. Bao, Relaxed conditions for convergence of batch BPAP for feedforward neural networks, *Neurocomputing*, vol.153, pp.174–179, 2015.

[41] Shao, Y.-H., Wang, Z., Yang, Z.-M., and N.-Y. Deng, Weighted linear loss support vector machine for large scale problems, *Procedia Computer Science*, vol.31, pp.639–647, 2014.

[42] Siniscalchi, S.M., Yu, D., Deng, L., and C.-H. Lee, Exploiting deep neural networks for detection-based speech recognition, *Neurocomputing*, vol.106, pp.148–157, 2013.

[43] Tsoi, A.C., and A. Back, Static and dynamic preprocessing methods in neural networks, *Engineering Applications of Artificial Intelligence*, vol.8, no.6, pp.633–642, 1995.

[44] Wang, L., and T. Chen, Multistability and complete convergence analysis on high-order neural networks with a class of nonsmooth activation functions, *Neurocomputing*, vol.152, pp.222–230, 2015.

[45] Wei, W., and Y. Xin, Rapid, man-made object morphological segmentation for aerial images using a multi-scaled, geometric image analysis, *Image and Vision Computing*, vol.28, no.4, pp.626–633, 2010.

[46] Wei, X., Phung, S.L., and A. Bouzerdoum, Object segmentation and classification using 3-D range camera, *Journal of Visual Communication and Image Representation*, vol.25, no.1, pp.74–85, 2014.

[47] Weng, J., Ahuja, N., and T.S. Huang, Learning recognition and segmentation using the cresceptron, *International Journal of Computer Vision*, vol.25, no.2, pp.109–143, 1997.

[48] Xu, D., Zhang, H., and D.P. Mandic, Convergence analysis of an augmented algorithm for fully complex-valued neural networks, *Neural Networks*, vol.69, pp.44–50, 2015.

[49] You, J., and H.A. Cohen, Classification and segmentation of rotated and scaled textured images using texture "tuned" masks, *Pattern Recognition*, vol.26, no.2, pp.245–258, 1993.

[50] Yu, C., Manry, M.T., Li, J., and P.L. Narasimha, An efficient hidden layer training method for the multilayer perceptron, *Neurocomputing*, vol.70, nos.1–3, pp.525–535, 2006.

[51] Zhang, Y., Li, X., Zhang, Z., Wu, F., and L. Zhao, Deep learning driven blockwise moving object detection with binary scene modeling, *Neurocomputing*, vol.168, pp.454–463, 2015.