Neural Comput & Applic (2012) 21:1783–1790
DOI 10.1007/s00521-011-0754-8

ORIGINAL ARTICLE

Influence of graphical weights' interpretation and filtration algorithms on generalization ability of neural networks applied to digit recognition

Maciej Kusy · Damian Szczepanski

Received: 9 November 2010 / Accepted: 22 October 2011 / Published online: 11 November 2011

Abstract In this paper, the method of the graphical interpretation of the single-layer network weights is introduced. It is shown that the network parameters can be converted to the image and their particular elements are the pixels. For this purpose, weight-to-pixel conversion formula is used. Moreover, new weights' modification method is proposed. The weight coefficients are computed on the basis of pixel values for which image filtration algorithms are implemented. The approach is applied to the weights of three types of the models: single-layer network, two-layer backpropagation network and the hybrid network. The performance of the models is then compared on two independent data sets. By means of the experiments, it is presented that the adjustment of the weights to new values decreases test error value compared to the error obtained for initial set of weights.

Keywords Weights · Neural network · Filtration · Digit recognition

1 Introduction

Character recognition problem has enjoyed great attention for a few decades. Back in 1972, the way of automatic recognition of handwritten characters was already described [1]. In mid-eighties, the method of "learning" character sets and various feature extraction techniques were proposed [2]. Machine learning techniques, such as neural networks, played a very important role in this domain [3]. In 1990, the application of a backpropagation neural network to the recognition of handwritten US Postal Service Office zip-code was presented [4]. Input data differed significantly in writing style, character size, overlapping numerals, postmarks, horizontal bars and marks on the envelope which made the recognition process more difficult. Even though, the performance on zip-code digits was 92% recognition, 1% substitution and 7% rejects [4]. In the field of character classification, one did not only focus on offline handwriting recognition, which is performed after the process of writing. The effort was also devoted to online, i.e. dynamic handwriting recognition in which the machine recognizes the characters while the user writes [5]. The transducers (converters, e.g. tablet) were used for this purpose, but the process was strongly dependent on the power of contemporary computers. It is necessary to emphasize that neural networks are not the only models that have been used in handwritten patterns classification. The other methods of computational intelligence have also been applied. There are many contributions that present the use of distance classifiers [6–8], support vector machines [9, 8] or decision trees [10].

In spite of the fact that the task of character recognition has been thoroughly explored, it still attracts a lot of researchers nowadays. A great number of scientists still apply neural networks for this purpose. The recognition of subcontinental languages, e.g. Chinese letters [11, 12], Persian fonts [13] or Indian numeral optical characters [8, 14, 15], receives an increasing attention. For these particular cases, backpropagation neural networks, particle swarm optimization neural networks, single-layer perceptrons and probabilistic neural networks were used. Numerous amount of work has been done on benchmarking Arabic digits (e.g. CENPARMI released by Concordia University, CEDAR released by CEDAR-SUNY Buffalo

M. Kusy (✉) · D. Szczepanski
Department of Electrical and Computer Engineering, Rzeszow University of Technology, Rzeszow, Poland
e-mail: mkusy@prz.edu.pl

or MNIST extracted from the NIST database) where various neural models (multilayer perceptrons, radial basis function networks, learning vector quantization networks and polynomial networks) were tested against state-of-the-art machine learning techniques such as nearest neighbor classifiers, naive Bayes, rule-based learning or support vector machines [16–18].

In this work, the concept of the graphical interpretation of the single-layer neural network weights is proposed. The model is designed to classify all digits; thus, it is equipped with 10 neurons where each element is responsible for the recognition of a single numeral. Once the training process of the network is completed, it is shown that it is possible to convert the weights to pixel values in order to transform model parameters into the images. On the basis of the fact that the networks weights can be regarded as an image, the filtration algorithms are applied to the pixels obtained from the weight values. The filtered pixels then serve for new weights computation. The idea is tested on two data sets using three types of the models: single-layer network, two-layer backpropagation network and the hybrid network by comparing the efficiency of the networks with the weights computed from the filtered images, and the performance of the models having original set of parameters. For computational purposes, all the models, image transformations and filtration algorithms were hard-coded in the authors' software.

The main motivation of such a research lies in the intention of understating how the neural model "perceives" input data, how it faces image filtration and whether it is capable of generalizing to unknown examples.

The paper is organized as follows. Section 2 describes handwritten digit data sets used for recognition. In Sect. 3, the neural networks employed to the classification are briefly described. Section 4 highlights the graphical interpretation of the single-layer network weights. Later on, in Sect. 5, the filtration algorithms applied to image pixels and new weight modification method are discussed. Section 6 verifies the performance of the neural networks in two digit classification tasks. Finally, Sect. 7 presents the conclusions.

2 Input data sets

Two digit databases are considered in the work. The first set represents the numerals entered by means of Wacom CTE-440/S graphics tablet. Its working area covered A6 letter format (127.6 × 92.8 mm). The device resolution reached 2,000 dpi (787 lines/cm). Input patterns were entered by means of wireless pencil lead. Total input data included 1,000 handwritten digits $(0, 1, \ldots, 9)$, which, in turn, were converted to 30 × 40 size. For the sake of unification of all digit patterns, some necessary transformation operations were carried out. Initially, all the digits needed to be rescaled [19] since while writing on the tablet, their size was different. Then, each image underwent binarization [20] to be converted from colorful to the one in gray scale. Additionally, in order to limit the information of the characters that were written with thick lines and to extract the parts that represent the relevant elements of the images, the patterns were peeled off using skeletonization algorithm [21], [22]. Finally, due to the fact that the placement of the digits within the frame of the device's screen was different, all the characters had to be centered for proper representation in 30 × 40 pixel pattern.

The second set was the MNIST database [7], a subset of a larger set available from National Institute of Standards and Technology (NIST). It consisted of 60,000 training examples and 10,000 test examples. The digits were size-normalized and centered in a fixed-size 28 × 28 image. The images contained gray levels as a result of the anti-aliasing technique used by the normalization algorithm. The regular 28 × 28 database along with the content description and a performance results for some computational intelligence methods is available at [23].

3 Neural networks used in digit recognition

Three types of neural networks were analyzed in the research: single-layer network, two-layer backpropagation network and the hybrid network. All the models are shortly discussed in the following subsections.

3.1 Single-layer network

The network consisted of one output layer and $k = 1, \ldots, n$ input elements, which represented image size of each pattern $\mathbf{x}_i = [x_{1i}, \ldots, x_{ni}]$ for $i = 1, \ldots, l$ where l is the total number of data set. All the examples, through the weight vector $\mathbf{w1}_j = [w1_{j1}, \ldots, w1_{jn}]$, were connected to j-th neuron of the output layer for $j = 1, \ldots, m$ where $m = 10$ is the number of classified digits $(0, 1, \ldots, 9)$. Each output neuron computed the weighted sum of the input signals, which was fed forward to get activated according to the formula:

$$a1_{ji} = f\left(\sum_{k=1}^{n} w1_{jk} \cdot x_{ki} + b1_j\right) \tag{1}$$

where $b1_j$ is the bias and $f(\cdot)$ is defined as log-sigmoid transfer function [24]. The output of each of all 10 neurons belongs to the interval (0, 1); therefore, in order to recognize the digit at the single presentation, the highest output activation was determined and the pattern was classified to the class corresponding to the strongest signal represented by the j-th neuron. The network was trained

using standard Widrow-Hoff gradient descent learning algorithm [25], which relied on the following weights' update:

$$w1_{jk}^{(\text{new})} = w1_{jk}^{(\text{old})} + \eta \cdot \left(t_{ji} - a1_{ji}\right) \cdot f' \cdot x_{ki} \tag{2}$$

where t_{ji} is an element of a target vector $\mathbf{t}_i$ given for input signal $\mathbf{x}_i$, η is positive learning rate and f' is the derivative of the transfer function.

3.2 Two-layer backpropagation network

The network was composed of one hidden layer having m neurons connected to n input elements, and the output layer of $o = 1, \ldots, p = 10$ units responsible for classification of a particular digit. All output neurons were fired using log-sigmoid transfer function f with the weighted sum of the hidden layer signals as the argument:

$$a2_{oi} = f\left(\sum_{j=1}^{m} w2_{oj} \cdot a1_{ji} + b2_o\right) \tag{3}$$

where $w2_{oj}$ is the element of the second layer weight vector $\mathbf{w2}_o = [w2_{o1}, \ldots, w2_{om}]$, $a1_{ji}$ is defined in (1) and $b2_o$ is the second layer bias. As in case of single-layer network, the highest output activation determined the class of a classified digit. The model was trained using standard backpropagation algorithm that amounted to the following update of the weights of the first and second layer:

$$w1_{ji}^{(\text{new})} = w1_{ji}^{(\text{old})} + \eta \cdot \delta_{ji} \cdot x_{ki} \tag{4}$$

$$w2_{oj}^{(\text{new})} = w2_{oj}^{(\text{old})} + \eta \cdot \delta_{oi} \cdot a1_{ji} \tag{5}$$

where $\delta_{ji} = \sum_{o=1}^{p} \left(\delta_{oi} \cdot w2_{oj}\right), \delta_{oi} = \left(t_{oi} - a2_{oi}\right) \cdot f' \cdot a1_{ji}$ and η is the learning rate [24].

3.3 Hybrid network

The network was built of two layers: self-organized Kohonen feature map [26] associated with a row input and a non-linear layer of 10 neurons. In the Kohonen layer, each of $j = 1, \ldots, m$ neurons computed the "distance" between the input vector $\mathbf{x}_i$ and the weight $\mathbf{w1}_j$ according to scalar product-based measure [24]:

$$n1_{ji} = d_j\left(\mathbf{x}_i, \mathbf{w1}_j\right) = \|\mathbf{x}_i\| \cdot \left\|\mathbf{w1}_j\right\| \cdot \cos\left(\mathbf{x}_i, \mathbf{w1}_j\right) \tag{6}$$

The d_j "distance" determined the winner-neuron along with its neighborhood. All the neurons in the layer were activated by means of radial basis transfer function [27]:

$$a1_{ji} = \exp\left(-\frac{\left|n1_{wi} - n1_{ji}\right|}{\sigma^2}\right) \tag{7}$$

where $n1_{wi}$ is the activation of the winner-neuron. In the output layer, each neuron calculated the weighted sum of $a1_{ji}$ signals which was fed as the argument to log-sigmoid activation function (3). The Kohonen layer was trained using neural gas algorithm [28]. The second layer was trained by means of standard gradient descent learning algorithm.

4 Graphical interpretation of single-layer network's weights

The initial idea behind the interpretation of the weights of the single-layer network was to understand how the particular neurons of the artificial model "see" their coefficients. Since the row input consists of 30×40 elements for the tablet data and 28×28 elements for the NIST database connected to $m = 10$ neurons, one may ponder whether it is possible to generate the pictures of the same size showing the weight values computed for each neuron after training process. Can the weights be perceived as an image of a digit?

In order to find the answer, it is necessary to provide the method of how to convert the weights (a set of real numbers) into the values that can correspond to the pixels with some brightness. Once such a transformation is determined, one can represent the set of calculated pixels in a resolution image. In this section, such a weight visualization is proposed. The following example highlights the idea.

Let us consider some network having three inputs and m outputs and assume the weighs for j-th neuron: $w_{j1} = 0.3$, $w_{j2} = -0.1$ and $w_{j3} = 0.7$, where $j = 1, \ldots, m$. Then, let us find the maximum and minimum of these coefficients, $w_{j\max} = 0.7$ and $w_{j\min} = -0.1$, in this case. Furthermore, let us convert the weight values using the following normalization:

$$p_{jk} = \left\lfloor 255 \cdot \frac{w_{jk} + \left|w_{j\min}\right|}{w_{j\max} - w_{j\min}} \right\rfloor \tag{8}$$

where $\lfloor x \rfloor$ is defined as the floor of some real number x. Now, for this particular set of weights, p_{jk} is called a pixel and formula (8) generates the image with the pixel values: $p_{j1} = 127$, $p_{j2} = 0$ and $p_{j3} = 255$ in the gray scale. Such an image has a color palette describing each pixel using a single byte. In this case, the image consisting of gray, black and white pixel is given, thus p_{jk} defines the brightness in the gray scale.

Such a transformation was applied to the set of weights calculated by the single-layer network designed to recognize digits of the tablet and NIST data set. Figure 1 represents ten pictures of the pixels generated from the weights using (8) for $m = 10$ neurons. In the upper row, 28×28 images of the weights for NIST database are shown. At the bottom, the pictures of the weights for the

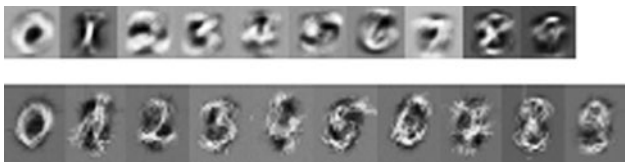

Fig. 1 Graphical presentation of the weights generated for all $m = 10$ output neurons of the single-layer network trained on 60,000 NIST digits with 28×28 resolution (*upper part images*) and 30×40 size 1,000 digits entered by means of Wacom CTE-440/S tablet (*lower part images*)

tablet data are presented. In both cases, the network was trained using Widrow-Hoff gradient descent algorithm (2) with $\eta = 0.6$. First top and bottom left images in Fig. 1 represent the weight values for the first neuron that learnt to recognize digit 0. The weights of remaining neurons, treated as the pixels in the image (seen as $1, 2, 3, \ldots, 9$), also resemble appropriate digits from the input data, i.e. $1, 2, 3, \ldots, 9$.

As shown, neural network parameters determined after training process do not have to be only treated as some numbers that allow the model to classify input examples. The weights can also be interpreted as the images and, what was shown in this section, such an interpretation can illustrate the effect of network's training and the way the artificial model "understands" digit recognition.

5 Image filtration and the weight adjustment

As shown in Sect. 4, the neural network weights can be interpreted in a graphical form. Each appropriately normalized coefficient is then treated as the element of the image seen by the neuron. This image, after the model training process, resembles a digit to a large degree. However, the picture is not so "perfect" as the original input pattern. For example, the neuron weights obtained from NIST digits illustrated in Fig. 1 in the form of pixels are blurry. The weights calculated from the tablet numerals, as the image, do not have a strong signal (white pixels) but are more distinct. For this reasons, high- and low-pass filtration algorithms were applied to pixels computed from the set of optimal weights found within the network training on tablet and NIST data sets. On the basis of filtered pixels, neural network weight modification was introduced. Following subsections describe the filtration algorithms and the solution of how to adjust the weights parameters on the basis of new pixel values.

5.1 Filtration algorithms

In the research, three methods of the filtration of the images representing neural networks weights were considered. The first approach amounted to the application of a high-pass filtration algorithm to the pixels. It was achieved by the use of HP3 mask defined as [19]:

$$\mathrm{HP3} = \begin{matrix} 0 & -1 & 0 \\ -1 & 20 & -1 \\ 0 & -1 & 0 \end{matrix} \tag{9}$$

Such a procedure passed and amplified the high-frequency elements (such as noise or edges) what resulted in sharpening the picture. The second solution relied on the use of low-pass filtration that passed and amplified low-frequency elements suppressing small group of noise and details at the same time. This method also smoothed and blurred the images. Two type of low-pass filters were used in the work: LP3 mask [19]:

$$\mathrm{LP3} = \begin{matrix} 1 & 1 & 1 \\ 1 & 12 & 1 \\ 1 & 1 & 1 \end{matrix} \tag{10}$$

and the Gaussian filter [19]:

$$\mathrm{LPG} = \begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix} \tag{11}$$

On the basis of a given filter and by positioning the mask on the pixels of an image, new pixel values were recomputed [29]:

$$\begin{aligned} p_{jk}^{(\mathrm{new})} = \frac{1}{s}\big(&M_{-1,-1}\cdot p_{j-1,k-1} + M_{0,-1}\cdot p_{j,k-1} + M_{1,-1}\cdot p_{j+1,k-1} \\ &+ M_{-1,0}\cdot p_{j-1,k} + M_{0,0}\cdot p_{j,k} + M_{1,0}\cdot p_{j+1,k} \\ &+ M_{-1,1}\cdot p_{j-1,k+1} + M_{0,1}\cdot p_{j,k+1} + M_{1,1}\cdot p_{j+1,k+1}\big) \end{aligned} \tag{12}$$

where p_{jk} is the source image pixel, s is the normalization parameter: $s = M_{-1,-1} + M_{0,-1} + M_{1,-1} + M_{-1,0} + M_{0,0} + M_{1,0} + M_{-1,1} + M_{0,1} + M_{1,1}$ and $M_{x,y}$ denotes one of 3×3 mask element defined in (9), (10) or (11).

5.2 Weights adjustment

Since new pixel values in particular images are normalized weight coefficients of the neural network, it is possible to find the inverse mapping of the weight-to-pixel transformation formula (8) that scales single picture elements into network parameters. It can be defined as follows:

$$w_{jk}^{(\mathrm{new})} = f^{-1}\left(p_{jk}^{(\mathrm{new})}\right) \approx \frac{p_{jk}^{(\mathrm{new})}}{255}\cdot\left(w_{j\max} - w_{j\min}\right) - \left|w_{j\min}\right| \tag{13}$$

where $p_{jk}^{(\mathrm{new})}$ is defined in (12). Depending on the mask applied, the pixels change their intensity getting more sharp or blurry what makes the weights in the image

behave in the similar way. Now, if one updates the neural network weights by means of (13) mapping, the model's classification ability should change. The next section presents the test error comparison between analyzed networks with original and modified set of weights in the classification of tablet and NIST data sets.

6 The performance of neural networks on digit data sets

In this part of work, the comparative efficiency analysis was conducted for the single-layer network, two-layer backpropagation network and the hybrid network in the classification of tablet and NIST data sets. The comparison was carried out by measuring two indicators of the performance: test error—calculated by the models with the weights obtained after training process, and test error after filtration—determined by the networks with the weights updated according to the mapping (13). Both factors were computed on the test set different from the patterns used in the training process: 200 of numerals for the tablet data set and 10,000 digits for NIST database. The errors were measured as the function of the network training parameters. Two following subsections highlight the results received on each data set. Afterward, a short summary is added.

6.1 Tablet data set

In the simulation, the following parameters of the networks were considered: learning rate $\eta = \{0.1, 0.2, 0.4, 0.6, 0.8, 0.9, 0.95, 0.99\}$ for single-layer network, and the number of hidden neurons for backpropagation network ($j = 5, \ldots, 30$) and the hybrid network ($j = \{50, 100, 200, 300, 400\}$). All models were trained on the same 800 training patterns and then tested on 200 independent samples. After the weights update, the models were re-tested on the same set of 200 digits. Table 1 presents the outcome of the comparison of the models in terms of the lowest percentage of test error and test errors after filtration. On the basis of the results, one can infer the following:

- Gaussian filtration (LPG mask) provided the lowest test error for single-layer network (10.425%), two-layer backpropagation network (9.468%) and the hybrid network (17.234%),
- the lowest overall test error (9.468%) was achieved by two-layer backpropagation network with the set of weights modified by Gaussian filtration algorithm,
- the highest reduction in the error rate was equalled 2.925%—it was found when applying Gaussian filtration to the weights of backpropagation network,
- all filtration algorithms decreased 19.894% test error by the margin of 1.276% (HP3), 2.021% (LP3) and 2.659% (LPG) for the hybrid network,
- HP3 mask filtration increased test error by 0.106% and 0.851% for the single-layer and backpropagation network, respectively.

As shown, for this particular data set, throughout the use of the filtration algorithm (LPG mask), it was possible to find a 23.61% gain in the test error rate of the model (backpropagation network).

6.2 NIST database

The same set of learning rate and hidden neuron values was applied in the classification NIST database to single-layer network and the hybrid network, respectively. For two-layer backpropagation network, the number of hidden neurons was taken from the interval: $j = 7, \ldots, 30$. The models were trained on 60,000 digits and then validated on the set of 1,000 numerals by computing test error and test errors after filtration. In Table 2, the lowest percentage for both indicators determined on NIST database for all models is presented. The results lead to the following conclusions:

- HP3 and LP3 mask decreased test set error for each neural network,
- all filtration algorithms decreased 18.174% test error by the margin of 0.424% (HP3), 0.382% (LP3) and 0.286% (LPG) for the hybrid network,
- LP3 mask filtration applied to backpropagation network weights provided the lowest test error among all classifiers,

Table 1 The lowest percentage of test error (Test) and test errors after filtration (HP3, LP3, LPG) found for single-layer network, two-layer backpropagation network and hybrid network in tablet digits classification

Model	Minimum error values [%]			
	Test	HP3	LP3	LPG
Single-layer network	12.127	12.234	11.117	10.425
Backpropagation network	12.394	13.245	10.585	9.468
Hybrid network	19.894	18.617	17.872	17.234

Table 2 The lowest percentage of test error (Test) and test errors after filtration (HP3, LP3, LPG) recorded for single-layer network, two-layer backpropagation network and hybrid network in NIST database pattern recognition

Model	Minimum error values [%]			
	Test	HP3	LP3	LPG
Single-layer network	9.376	9.172	9.294	9.488
Backpropagation network	9.078	9.068	8.966	9.078
Hybrid network	18.174	17.750	17.792	17.888

- Gaussian filtration algorithm made the test error increase for the single-layer network by 0.111%.

NIST digits have frequently been referred as the testing base for various classification algorithms. For example, in [7], the authors obtained quite remarkable results for two-layer network getting 4.7% of the test error rate. However, it was achieved by the use of 300 hidden neurons for the model.

6.3 Summary

This subsection illustrates how the performance of the networks evolved along with model parameter changes. Figures 2, 3 and 4 show the plots of percentage of misclassified test digits as a function of model parameter for single-layer network, two-layer backpropagation network and the hybrid network, respectively. The test error (diamond marker) and the test errors after filtration (square, triangle and circle) are set together for a better comparison. Dash and solid lines represent the errors computed on the tablet and NIST data set, respectively. While analyzing depicted dependencies, one can observe that for the tablet data classification, LP3 and LPG test errors were always smaller than the test error computed on the original set of weights for each neural network within a whole range of model parameters. HP3 filtration error, in turn, fell below test error value only in two cases: for the single-layer network ($\eta = \{0.1,\ 0.2\}$) and the hybrid network (300 hidden neurons). This phenomenon can be explained by the rule that HP3 mask strengthened the details of the images what weakened general features of the numerals. Therefore, LP3 and LPG masks, by blurring the digit images, increased generalization ability of the networks.

However, HP3 mask applied to the weights obtained from the training of the NIST digits made the filtration test error lower than the error determined on the unfiltered coefficients for the single-layer network and the hybrid network. LPG filtration, as shown in Figs. 2 and 3 provided here worse results for the single- and two-layer networks, respectively. It can be justified by the fact that the set of weights as the image was blurry (Fig. 1). On the other hand, for the hybrid network (Fig. 4), all filtration algorithms decreased test error rate for each number of hidden neurons.

7 Conclusion

In the article, the method of single-layer neural network weights interpretation was proposed. The network was destined to recognize digits from the range $0, 1, \ldots, 9$; therefore, it was built of 10 neurons. Each unit recognized single numeral. It was shown that after the training of the model, one is possible to transform the weight values to the

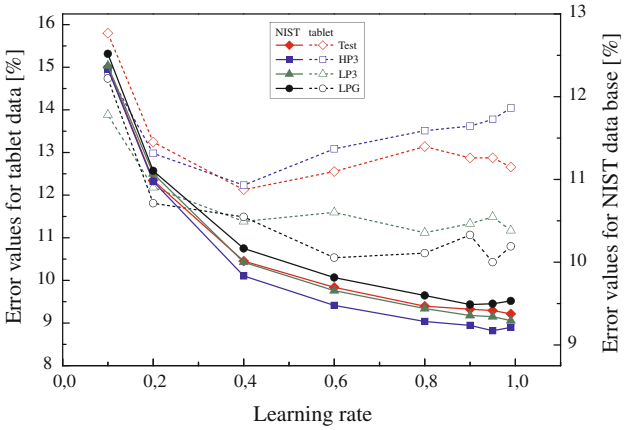

Fig. 2 Test and test after filtration errors plotted as the function of learning rate parameter for the single-layer network in the classification of the tablet (*dash*, *left axis*) and NIST (*solid*, *right axis*) digits

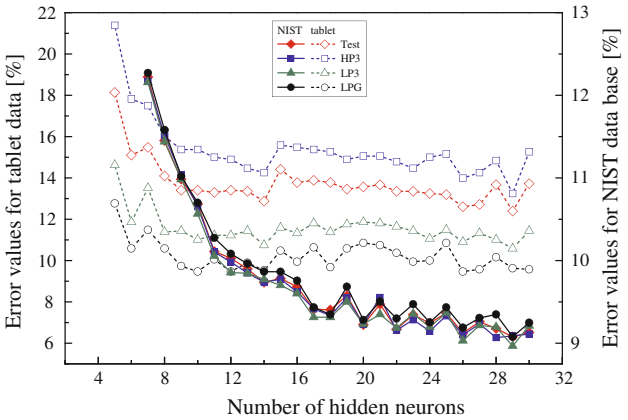

Fig. 3 Test and test after filtration errors plotted as the function of hidden neurons number for two-layer backpropagation network in the classification of the tablet (*dash*, *left axis*) and NIST (*solid*, *right axis*) digits

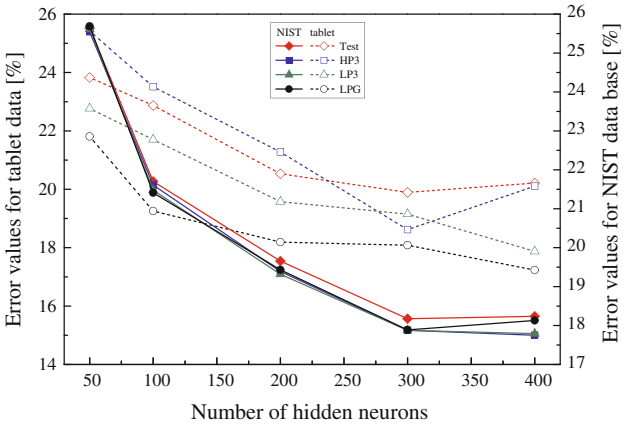

Fig. 4 Test and test after filtration errors plotted as the function of hidden neurons number for the hybrid network in the classification of the tablet (*dash*, *left axis*) and NIST (*solid*, *right axis*) digits

image pixels. The idea was tested on two data sets: 30×40 resolution 1,000 digits entered by means of the graphical tablet and 60,000 NIST web page database digits with 28×28 size. Furthermore, high- and low-pass filtration algorithms were applied to the pixels computed from the weight values. On the basis of the filtered pixels, the weights of the network were adjusted. This approach was then verified on three types of the models: single-layer network, two-layer backpropagation network and the hybrid network by comparing the performance of the models having the weights computed from the filtered images and the coefficients obtained after training process. The analysis were carried out on both data sets. The results presented in the work showed that, in both data classification cases, the filtration algorithms decreased test error calculated by the networks with the weights set to values determined after training process. In particular, in tablet data recognition, the use of the LPG mask (Gaussian low-pass filter) provided the lowest test error for single-layer network (10.425%), two-layer backpropagation network (9.468%) and the hybrid network (17.234%). Moreover, this filtration algorithm applied to the weights of back-propagation network reduced the test error rate by the margin of 2.925% what yielded the lowest test error among all models (9.468%).

The improvement obtained by considered networks in the test error rate for NIST digit recognition was not that large though. The highest reduction of this indicator (0.424%) was obtained when using high-pass filtration (HP3 mask) to the weights of the hybrid network. The application of both HP3 and LP3 masks decreased admittedly the test error value, but the gain was subtle. It can be explained by the fact that this particular data set is a web base to which no image preprocessing was applied. In contrast, the tablet data set images, before been fed as the input to the networks, underwent the skeletonization process that extracted the shape of pattern digits.

The entire process of computing the pixels from the optimal network weights, applying the filtration algorithm to calculated pixels and, finally, updating the weights on the basis of filtered pixels can increase the generalization ability of the neural network. However, it is important to add that such an improvement can be found if an appropriate image filtration is applied. Sometimes, it may even amount to a trial and error approach. Moreover, some data preprocessing has to be performed since the images to be classified usually contain a lot of information, which mislead the network in the process of generalization.

Acknowledgments This research was partially supported by Rzeszow University of Technology Grant No. U-8255/DS and NN 514 705540 from National Science Centre. The authors are grateful to valuable comments of the anonymous reviewer. All the remarks significantly improved the quality of the manuscript.

References

1. Harmon LD (1972) Automatic recognition of print and script. Proc IEEE 60(10):1165–1176
2. Davis RH, Lyall J (1986) Recognition of handwritten characters - a review. J Image Vis Comput 4(4):208–218
3. Bishop M (1995) Neural networks for pattern recognition. Oxford University Press, New York
4. LeCun Y, Matan O, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jacket LD, Baird HS (1990) Handwritten zip code recognition with multilayer networks. In: Proceedings of 10th international conference on pattern recognition, vol 2, pp 35–40, Atlantic City, USA
5. Tappert CC, Suen CY, Wakahara T (1990) The state of the art in on-line handwriting recognition. IEEE Trans Pattern Anal Mach Intell 12(8):787–808
6. Weideman WE, Manry MT, Yau HC, Gong W (1995) Comparisons of a Neural Network and a Nearest-Neighbor Classifier via the Numeric Handprint Recognition Problem. IEEE Trans Neural Net 6(6):1524–1530
7. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11): 2278–2324
8. Shrivastava SK, Gharde SS (2010) Support Vector Machine for Handwritten Devanagari Numeral Recognition. Int J Comput Appl 7(11):9–14
9. Vapnik VN (1998) Statistical learning theory. Wiley, New York
10. Wen-Li J, Zheng-Xing S, Bo Y, Wen-Tao Zheng, Wen-Hui X (2006) User-independent online handwritten digit recognition. In: International conference on machine learning and cybernetics, pp 3359–3364, Dalian, China
11. Cheng-Lin L, Jaeger S, Nakagawa M (2004) Online recognition of Chinese characters: the state-of-the-art. IEEE Trans Pattern Anal Mach Intell 26(2):198–203
12. Zhitao G, Jinli Y, Yongfeng D, Junhua G (2009) Handwritten Chinese characters recognition based on PSO neural networks. In: Second international conference on intelligent networks and intelligent systems, pp 350–353
13. Pourmohammad A, Ahadi SM (2009) Using single-layer neural network for recognition of isolated handwritten Persian digits. In: 7th International conference on Information. Communicat Sig Proc, pp 1–4, Macau
14. Al-Omari FA, Al-Jarrah O (2004) Handwritten Indian numerals recognition system using probabilistic neural networks. Adv Eng Inform 18:9–16
15. Desai AA (2010) Gujarati handwritten numeral optical character reorganization through neural network. Pattern Recogn 43:2582–2589
16. Liu C, Nakashima K, Sako H, Fujisawa H (2003) Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recogn 36:2271–2285
17. Al-Omari S, Sumari P, Al-Taweel SA, Husain AJA (2009) Digital recognition using neural network. J Comput Sci 5(6):427–434
18. El-Alfy EM (2010) Offline recognition of handwritten numeral characters with polynomial neural networks using topological features. Lect Notes Comput Sci 6085:173–183
19. Watkins CD, Sadun A, Marenka S (1995) Nowoczesne metody przetwarzania obrazu. WNT, Warszawa

20. Trier OD, Taxt T (1995) Evaluation of binarization methods for document images. IEEE Trans Pattern Anal Mach Intell 17(3): 312–315
21. Gonzales R, Woods RE (2002) Digital image processing. Prentice Hall, New Jersey
22. Gupta R, Kaur R (2008) Skeletonization algorithm for numeral patterns. Int J Sig Proc Image Proc Pattern Recogn 63–72
23. Lecun Y. NIST data base repository. http://yann.lecun.com/exdb/mnist/
24. Tadeusiewicz R (1993) Sieci neuronowe. Akademicka Oficyna Wydawnicza, Warszawa
25. Widrow B, Hoff ME (1960) Adaptive switching circuits. IRE WESCON Conv Rec 4:96–104
26. Kohonen T (1995) Self-organizing maps. Springer, Berlin
27. Osowski S (2006) Sieci neuronowe do przetwarzania informacji. WNT, Warszawa
28. Martinetz M, Berkovich S, Schulten K (1993) Neural-gas network for vector quantization and its application to time-series prediction. IEEE Trans Neural Net 4(4):558–569
29. Tadeusiewicz R, Korohoda P (1997) Algorytmy i metody komputerowej analizy i przetwarzania obrazow. Poldex, Krakow