REVIEW AND ANALYSIS OF HIDDEN NEURON NUMBER EFFECT OF SHALLOW BACKPROPAGATION NEURAL NETWORKS

B. Sekeroglu, *K. Dimililer*†

Abstract: Shallow neural network implementations are still popular for real-life classification problems that require rapid achievements with limited data. Parameters selection such as hidden neuron number, learning rate and momentum factor of neural networks are the main challenges that causes time loss during these implementations. In these parameters, the determination of hidden neuron numbers is the main drawback that affects both training and generalization phases of any neural system for learning efficiency and system accuracy. In this study, several experiments are performed in order to observe the effect of hidden neuron number of 3-layered backpropagation neural network on the generalization rate of classification problems using both numerical datasets and image databases. Experiments are performed by considering the increasing number of total processing elements, and various numbers of hidden neurons are used during the training. The results of each hidden neuron number are analyzed according to the accuracy rates and iteration numbers during the convergence. Results show that the effect of the hidden neuron numbers mainly depends on the number of training patterns. Also obtained results suggest intervals of hidden neuron numbers for different number of total processing elements and training patterns.

Key words: *backpropagation neural networks, hidden neuron number, training patterns, total processing elements*

Received: March 19, 2019 **DOI:** 10.14311/NNW.2020.30.008
Revised and accepted: April 30, 2020

1. Introduction

Neural network (NN) is an applicable artificial intelligence (AI) field that can be used effectively in many pattern recognition applications as supervised or unsupervised [1]. Flexible outputs of NN and capability of solving non-linear tasks, produce successful and considerable results [1]. Thus, achieved results become more accurate for real-life applications.

*Boran Sekeroglu – Corresponding author; Information Systems Engineering Department of the Near East University, Nicosia, TRNC, Turkey, E-mail: boran.sekeroglu@neu.edu.tr

†Kamil Dimililer; Electrical and Electronic Engineering Department of the Near East University, Nicosia, TRNC, Turkey, E-mail: kamil.dimililer@neu.edu.tr

One of the most common and popular supervised learning algorithms is the backpropagation neural network (BPNN) [2]. BPNN is the most frequently used learning algorithm in shallow neural networks because of its simplicity in the implementation, and efficiency of the obtained results in real life applications. Diversity of implementations used BPNN effectively in classification [3, 4], control [5], prediction [6] and optimization [7] problems.

However, several in-explicit factors such as the number of hidden layers, the number of the neurons in these layers, the learning rate, and momentum factors are still determined by trial and error, dependent on the application, by considering the convergence efficiency and generalization results. The learning rate and momentum factor are used to tune the adjustments for each iteration and to avoid local minima during the convergence respectively [8]. Lower value learning rates and relatively higher value momentum rates were preferred to be used in BPNN. The usage of multi-hidden layers in BPNN increases the computational cost but rarely improves the classification rates [8]. Thus, researchers were redirected to use a single hidden layer in BPNN implementations. However, at that time, the determination of hidden neuron numbers which is the most important uncertainty emerged. For this reason, several experiments had been performed and different approaches as constructive and pruning [9], were proposed by the researchers to determine the optimum hidden neuron number for a specific application.

Constructive approaches [10, 11] are based on adding extra neurons into the hidden layer which starts with an undersized number of hidden neurons until the satisfactory results obtained. Conversely, pruning approaches [12–14] remove the less relevant neurons start in an over-sized network to find the smallest optimum number of hidden neurons. Pruning approaches determine hidden neuron to be pruned according to the neuron activities with little influence [15]. The calculation of the influence of neurons can differ from each method and can be calculated by considering various information such as variance analysis of sensitivity information [13], quantified sensitivity measure [9], hidden neurons with zero values [16], apparent rate error [17], weight variation information [18], during the training.

In the literature, rare analysis was performed for hidden neuron number effect of backpropagation neural network. Shafi et al. [19] investigated hidden neuron and layer number effect of backpropagation architecture for a time frequency application. They concluded that, 40 hidden neuron number in a single hidden layer performed optimum result for their application and also it was mentioned that the increment of hidden neuron number decreases the success of neural network.

Researches show that the performed experiments to determine the hidden neuron number can produce different results for different applications. However, diversity of the methods for different approaches and the difficulties in the implementation of these methods cause extra time cost for the applications need rapid achievements. The analysis of the behavior or response of the neural system in terms of learning (convergence) efficiency and especially in generalization ability which is the main indicator of the success of the neural system, is required. Training and generalization of NN using diverse conditions with different number of hidden neurons is important in order to determine over-sized and undersized hidden neuron numbers. Over-sized number of hidden neurons causes over-fit the training data which models noise and this reduces the generalization ability of the network even the convergence is extremely successful [8]. The usage of undersized number of hidden neurons prevents the convergence of training data, therefore this may lead to obtain inefficient classification rates in generalization. However, the determination of these under or over-sized numbers of the hidden neurons is related to the complexity of the application, hence the neural network.

It is obvious that, this complexity of the NN should be considered as a structure by the inclusion of each element of NN as the number of training patterns and input neurons. The number of input and output neurons, which depends on the application, and the number of training patterns increase the complexity of the neural network, thus change the effect of the determined hidden neuron number during the training and generalization. In this study, the complexity of NN is called Total Processing Elements (TPE) of the neural system. During the increment of TPE, it is not known how much the number of hidden neurons needs to be increased or decreased. Therefore, various number of hidden neurons should be considered during the learning, and generalization rates should be analyzed for considered hidden neuron numbers in order to determine the interval of minimum and maximum hidden neuron numbers suitable for all applications. In this study, fourteen classification applications were performed with different number of hidden neurons in 3-layered network architecture.

The aim of this research was to analyze the behavior of BPNN by considering training iterations and foremost generalization rates and to contribute BPNN implementations by recommending general minimum and maximum hidden neuron numbers according to TPE of the applications to minimize time loss during trial and error.

The rest of the paper is organized as follows; Section 2 summarizes the several real-life applications using BPNN and considered hidden neuron numbers in these applications. Section 3 explains specifications of experimental design with considered numerical datasets and image databases and Section 4 presents experimental results. Discussions of experimental results and suggestions are placed in Section 5. Finally, Section 6 concludes the analyzes of results obtained in this research.

2. Hidden neuron usage in the literature

Several and different real-life applications and experiments were performed using BPNN, and various hidden neuron numbers were considered to be used. Different applications with various datasets or image databases may require the consideration of a different number of hidden neuron numbers, while the datasets strongly affects the convergence of BPNN. In this section, a summary of recent and varied real-life experiments, and considered hidden neuron numbers in these researches will be presented.

Recently, Li [20] proposed an automatic impedance matching method based on BPNN, and Liu [21] used BPNN to analyze high-power LED photoelectrothermal. Linlin [22] propose 3D indoor localization by implementing BPNN. Lilik [23] and Dimililer [24] used it to detect the lung cancer on CT scan images. Comparison of workload prediction model for cloud computing by Kumar et al. [6], privacy-preserving using modified BPNN for cloud computing by Yuan et al. [25], and a system in order to detect the sub-pixel land change for remotely sensed images by

Wu et al. [26] proposed using BPNN. Also, nonlinearity compensation of a photonic transducer-based optical current sensor in [27] and classification of four varieties of bulk rice grain images in [3] used BPNN. Nuclear power plant transients identifier [28], and optimal control for robotic arms [5] were implemented. In addition to these researches, Khashman and Sekeroglu [7] suggested BPNN to determine the optimum threshold value for document image enhancement. Dimililer [4] classified pest insects, and Khashman et al. [29] solved the classification problems of 2 Euro and 1 TL coins in slot machines using BPNN. Beside these applications, image compression [30], microRNA analysis [31] and medical applications [32] considered BPNN for performance improvement. Tab. I summarizes the applications and decided on hidden neuron numbers to be used.

Author	Year	Hidden Neuron Number
Yang Li et al.	2018	33
Hogwei Liu et al.	2017	20
Linlin et al.	2017	11
Kumar and Singh	2017	7
Dimililer and Zarrouk	2017	40
Wu et al.	2017	15
Wei et al.	2016	40
Dimililer et al.	2016	25
Yuan and Yu	2014	5
Yuan and Yu	2014	12
Yuan and Yu	2014	15
M-Bakhshayesh and Ghofrani	2014	10
Rubio	2012	20
Adali and Sekeroglu	2012	20

Tab. I *Hidden neuron numbers in recent applications.*

3. Experimental design

The complexity of NN may differ the effect of hidden neuron number, thus, a variety of NN architectures are needed to be considered in order to analyze the effect of hidden neuron number on the generalization ability and learning efficiency of NN. Over or undersized number of inputs and outputs which depend on the application, can be considered separately however, further training patterns should also be included for the calculation of the complexity of NN. Slight complexity of NN may cause smooth convergence of it and makes analyses of the effect of hidden neuron numbers on huge NN architecture, difficult. Therefore, besides low or high input neuron numbers, it is necessary to complicate the learning of the NN by using further training patterns, and it is called Total Processing Elements (TPE) of NN. TPE can be obtained by multiplication of the number of input neurons (number of attributes) and training patterns (instances for training) as shown in Eq. 1.

$$\text{TPE} = \text{In} \times \text{TrP}, \tag{1}$$

where In is the number of input neurons and TrP is the number of training patterns for the corresponding experiment.

Besides these, during the experimental design, lower instances with high training patterns and high instances with low training patterns were also considered in order to observe and analyse the effect of hidden neuron numbers in various conditions. It is clear that the characteristics and the selection of training patterns are also important for the NN during the learning and generalization phases. Obviously, classification problems are not limited only with image-based databases; thus numerical datasets that contain different acquired signals or attributes should be considered. In order to provide different and various classification experiments with varied patterns and TPE, six numerical datasets and three image databases that were used for the classification problems, were considered. Datasets are benchmark and frequently used numerical datasets; Iris [33], Wines [34], Hepatitis [35, 36], Sonar [37], Autism screening adult [38] and Cardiotocography [39], and the image databases are AT&T Face Database [40], CASIA Multi-Spectral Palmprint Database [41] and Coins Database [29].

Datasets were directly considered for training and generalization phases with normalized values, but image databases were used with different dimensions which represent input neuron numbers, and low or high training patterns to increase TPE.

Experiments were divided into five categories according to TPE; low-processing experiments (LPEX), medium-processing experiments (MPEX), standard-processing experiments (SPEX), high-processing experiments (HPEX) and over-sized processing experiments (OPEX). Tab. II shows the minimum and maximum TPE for each experimental category.

Category	Minimum TPE	Maximum TPE
LPEX	60	1000
MPEX	1000	10,000
SPEX	10,000	150,000
HPEX	150,000	500,000
OPEX	500,000	–

Tab. II *TPE values for experiment categories.*

Since the improvement of recognition rates of NN for any dataset or database was not the scope of this research, training patterns were randomly selected both for numerical datasets and image databases. To vary the number of training patterns, 10-30 % of total considered data were used for training. The rest of the patterns were used for the generalization phases of the experiments. Tab. III shows details of experiments with input and output neurons of NN, training patterns, and TPE of each experiment. In image databases, Experiment 2.1 (AT&T Face Database), Experiment 2.5 (Coins Database) and Experiment 3.1 (Casia Palmprint Database) consist of image pre-processing step with Average Pixel per Node (APPN) [29, 42] approach to reduce input data and observe the effect of hidden neuron number for image database with lower TPE. APPN is based on dividing an image into pre-defined segments and calculating the gray level average of the pixels belong to the corresponding segment. Thus, individual value for each segment was obtained and the input neuron number of NN was statistically reduced.

Data Name	Category	Exp. No.	Inp. N. Number	Output	Tr. Pat.	TPE	Type
Iris	LPEX	1.1	4	3	15	60	Numeric
Hepatit	LPEX	1.2	19	2	20	380	Numeric
Wines	LPEX	1.3	13	3	48	624	Numeric
AT&T Face	MPEX	2.1	100	10	30	3000	Image
Sonar	MPEX	2.2	60	2	60	3600	Numeric
CTG	MPEX	2.3	21	3	200	4200	Numeric
Autism	MPEX	2.4	16	2	280	4480	Numeric
Coins	MPEX	2.5	256	2	32	8092	Image
Casia Palmprint	SPEX	3.1	256	20	240	61,440	Image
Coins	SPEX	3.2	4096	2	16	65,536	Image
Coins	SPEX	3.3	4096	2	32	131,072	Image
Coins	HPEX	4.1	16,384	2	16	262,144	Image
AT&T Face	HPEX	4.2	10,000	10	30	300,000	Image
Casia Palmprint	OPEX	5.0	4096	20	240	983,040	Image

Tab. III *Details of experiments.*

As it is shown in Tab. I, researchers generally optimized hidden neuron numbers between 5-30 in performed experiments for specific applications. Thus, in this research, hidden neuron numbers are determined by starting from 5 to 30 with an increment of 5 hidden neurons for each training. However, the usage of more than 30 hidden neurons should also be analyzed in order to observe the effect on training and generalization phases with varied TPE. Therefore; 50, 75, 100, 150, and 200 hidden neurons are also considered in this research which can be described as over-sized hidden neuron numbers.

In following sections, NN architectures of experiments are shown as IN-HN-OUT, where *IN* and *OUT* is the number of input and output neurons which depend on the application, and *HN* is considered hidden neuron numbers which are 5, 10, 15, 20, 25, 30, 50, 75, 100, 150 and 200 in this research.

4. Experimental results

In this paper, six numerical datasets and three image databases were considered in fourteen experiments with varied values of training patterns and TPE. In each experiment, patterns were trained using eleven different hidden neuron numbers. After performing several preliminary experiments, the learning rate and momentum factor set to 0.00079 and 0.90 respectively which were the most efficient relationship between them for all experiments commonly. Root Mean Square (RMS) Error was used as stopping criteria of BPNN and it was determined as 0.003. Initial weights were assigned randomly between -0.30 and 0.30, and the Sigmoid activation function was used for all layers. Bias weights were also considered for each layer and tolerance value of BPNN was set to 0.5 which means any actual output which is less than 0.5 was considered as "unrecognized". While the effect of hidden neuron numbers on the performance was the objective of the study, fixed training and testing sets were considered for each experiment with different hidden neuron numbers, and experiments were performed using Hold-out Method, which is based on dividing dataset into two sets as training and testing, and obtaining results in one-run. The evaluation was performed using the accuracy rates obtained for each hidden neuron number on a fixed testing and training sets using fixed parameters.

Learning efficiency of patterns was analyzed by using training iterations for each hidden neuron number, and the generalization ability of BPNN which was the main indicator of the success of the neural system, was analyzed by calculating the accuracy of the system. Accuracy was calculated by dividing correctly classified untrained patterns to total untrained patterns which are considered in generalization.

As it is expected, test results of training data were obtained 100 % for all experiments and hidden neuron numbers.

4.1 Results of low-processing experiments

Three different numerical datasets; Iris, Hepatitis, and Wines were used in LPEX, and TPE is below 1000 for each experiment.

In Experiment 1.1 (Iris Dataset), 15 training and 51 test patterns were considered in 4-HN-3 architecture, and generalization ability did not differ with hidden neuron numbers from 5 to 30 and 94.11 % of accuracy was achieved. Defined RMS Error was reached by 15 hidden neuron numbers in minimum iterations and the usage of more hidden neurons increased the iterations linearly. 7.62 % of difference occurred between the maximum and minimum iteration numbers of 5 to 30 hidden neurons, however by considering all hidden neuron numbers, this difference increased to 31.50 % between the minimum and maximum iterations.

In Experiment 1.2 (Hepatitis Dataset) 20 training and 120 test patterns were considered in 19-HN-2 architecture. Accuracy differed unsteadily and independently to hidden neuron numbers and maximum is achieved by 75 hidden neuron numbers. However, only three more test patterns are recognized correctly in 75 hidden neuron numbers by comparing the minimum accuracy in this experiment. Minimum and maximum iteration numbers are achieved in 5 and 200 hidden neurons respectively. The difference between minimum and maximum iterations is 14.55 %.

In Experiment 1.3 (Wines Dataset), which was the last, but not the least experiment of LPEX, 48 training and 129 test patterns were considered in 13-HN-3 architecture. Hidden neurons from 5 to 75 produced same and highest accuracy by 96.15 % and hidden neuron numbers above 75 started to decrease accuracy but with only a single test pattern. Similar to the second experiment, minimum and maximum iteration numbers were achieved in 5 and 200 hidden neurons, respectively, and the difference between the minimum and maximum iterations was calculated as 24.81 %.

Tab. IV shows the obtained results of Low-Processing Experiments for all hidden neuron numbers.

Hidden Neuron No.	Experiment 1.1 Iterations	Experiment 1.1 Gen. Ability [%]	Experiment 1.2 Iterations	Experiment 1.2 Gen. Ability [%]	Experiment 1.3 Iterations	Experiment 1.3 Gen. Ability [%]
5	31,717	**94.11**	11,996	58.33	11,230	**96.15**
10	31,969	**94.11**	11,498	59.16	10,058	**96.15**
15	**29,816**	**94.11**	11,281	58.33	9,790	**96.15**
20	30,694	**94.11**	11,055	58.33	9,603	**96.15**
25	30,155	**94.11**	11,075	58.33	9,557	**96.15**
30	32,276	**94.11**	11,123	60.00	9,293	**96.15**
50	32,602	92.15	10,905	60.00	9,340	**96.15**
75	33,939	92.15	10,702	**60.83**	9,100	**96.15**
100	34,848	90.19	10,730	60.00	8,880	95.38
150	38,125	88.23	10,498	58.33	8,586	95.38
200	43,532	88.23	**10,250**	58.33	**8,444**	95.38

Tab. IV *Results of low-processing experiments.*

4.2 Results of medium-processing experiments

In MPEX, five experiments were performed using both numerical datasets and image databases. In these experiments, minimum and maximum TPE was 3000 and 8192, respectively.

In Experiment 2.1 that APPN was applied to face images, 30 train and 70 test images were considered in 100-HN-10 architecture. Even the convergence occurred in maximum iterations, the highest accuracy was achieved with 5 hidden neurons by 95.71 % and the difference between the highest and the closest one was 10 % (150 and 200 hidden neurons). Minimum and maximum iterations were recorded in 200 and 5 hidden neurons, respectively, with the difference of 84.32 %.

In Experiment 2.2 (Sonar Dataset), 60 train and 251 test data were used in 60-HN-2 architecture, and close results were obtained by 69.13 % and 69.75 % with 5 and 200 hidden neurons, only by 1 test pattern difference. Even the close iterations were computed, minimum and maximum iterations occurred in 200 and 5 hidden neurons with 8.62 % of difference.

In Experiment 2.3 (CTG Dataset), 21-HN-3 architecture with 200 training and 1926 test patterns were used. The highest and lowest accuracy were obtained with 75 hidden neurons and 5 hidden neurons with classification of 19 more test patterns correctly. Minimum and maximum iterations were recorded in 10 and 150 with 10.11 % of difference.

In Experiment 2.4 (Autism Dataset), 16-HN-2 architecture with 280 training and 422 test patterns and, close and similar results are obtained in the generalization. However, the highest accuracy is achieved with 150 and 200 hidden neurons by 99.52 %. 16-100-2 architecture is converged in minimum iterations and 16-150-2 is converged in maximum iterations with a difference of 68.05 %.

In Experiment 2.5 (Coins Database), 256-HN-2 architecture is used and APPN applied to coin images. Totally 32 training and 48 testing images were considered in this experiment. 5 hidden neurons achieved 90.63 % of accuracy and the rest of the hidden neurons produced 75 % of accuracy. Similar to Experiment 2.1 and 2.2, lowest and highest hidden neuron numbers reached to RMS Error in maximum and minimum iterations with a difference of 55.25 %.

Tab. V shows the obtained results of Medium-Processing Experiments for all hidden neuron numbers.

	Exp. 2.1		Exp. 2.2		Exp. 2.3		Exp. 2.4		Exp. 2.5	
Hid. N. No.	Iter.	Gen. Ab. [%]	Iter.	Gen. Ab. [%]	Iter.	Gen. Ab. [%]	Iter.	Gen. Ab. [%]	Iter.	Gen. Ab. [%]
5	30,294	**95.71**	9,020	69.13	6,629	62.70	1,998	99.05	6,959	**90.73**
10	12,246	80.00	8,935	67.90	**5,980**	63.37	2,041	99.05	5,036	75.00
15	9,228	84.28	8,854	67.90	6,159	63.37	1,953	99.29	5,341	75.00
20	8,433	82.85	8,929	67.90	6,188	63.27	1,974	99.05	4,262	75.00
25	8,184	88.57	8,701	67.90	6,315	63.27	1,910	99.29	4,181	75.00
30	7,492	82.85	8,763	67.90	6,161	63.53	1,939	99.29	4,071	75.00
50	6,542	84.28	8,764	67.28	6,177	63.63	1,865	99.05	3,835	75.00
75	5,916	84.28	8,554	68.51	6,012	**63.68**	1,817	99.29	3,486	75.00
100	5,397	84.28	8,488	68.51	6,157	63.48	**1,706**	99.05	3,484	75.00
150	4,916	85.71	8,438	68.51	6,653	63.53	5,341	**99.52**	3,197	75.00
200	**4,748**	85.71	**8,242**	**69.75**	6,132	63.53	2,591	**99.52**	**3,113**	75.00

Tab. V *Results of medium-processing experiments.*

4.3 Results of standard-processing experiments

Three experiments were performed in SPEX using Palmprint and Coins databases where the TPEs were between 61,440 and 131,072.

In Experiment 3.1, 256-HN-20 architecture was used and APPN is applied to Palmprint images to reduce the inputs of BPNN. 240 training and 1200 test images were considered to identify 20 persons using palm prints. Lowest and highest accuracy was obtained using 5 and 25 hidden neurons with 13.58 % and 30.41 %, respectively. BPNN could not converge using 75 and more hidden neurons. 5 and 25 hidden neurons converged in maximum and minimum iterations with a difference of 94.87 %.

In Experiment 3.2, which the coins images were considered without any pre-processing, 4096-HN-2 architecture was used with 16 training and 64 test patterns. 20,100 and 150 hidden neurons produced the lowest accuracy with one more mistake than the rest of the hidden neuron numbers which achieved the highest one by 89.33 %. After linear decrements of iteration numbers, 5 and 200 hidden neurons reached RMS Error in maximum and minimum iterations.

Similar to Experiment 3.2, coins images were used in Experiment 3.3 with the same image dimensions, yet input neuron number but with 32 training and 48 testing patterns. All hidden neuron numbers achieved the same accuracy by 75.00 %. Even the recognition rates differed in Experiment 3.2 and 3.3, minimum and maximum iterations and change of iterations were similar in 200 and 5 hidden neurons with 87.30 %.

Tab. VI shows the obtained results of Standard-Processing Experiments for all hidden neuron numbers.

	Experiment 3.1		Experiment 3.2		Experiment 3.3	
Hidden Neuron No.	Iterations	Gen. Ability [%]	Iterations	Gen. Ability [%]	Iterations	Gen. Ability [%]
5	311,007	13.58	8,287	**83.33**	4,191	**75.00**
10	20,904	29.00	4,432	**83.33**	3,543	**75.00**
15	18,731	27.41	3,092	**83.33**	1,794	**75.00**
20	35,114	28.33	2,811	81.25	1,209	**75.00**
25	**15,939**	**30.41**	1,965	**83.33**	1,811	**75.00**
30	20,602	29.83	11,123	**83.33**	1,121	**75.00**
50	63,499	28.75	10,905	**83.33**	748	**75.00**
75	not converged		1,381	**83.33**	817	**75.00**
100	not converged		956	81.25	599	**75.00**
150	not converged		1,084	81.25	631	**75.00**
200	not converged		**884**	**83.33**	**532**	**75.00**

Tab. VI *Results of standard-processing experiments.*

4.4 Results of high-processing experiments

In HPEX, Coins and Face databases were used with 262,144 and 300,000 TPE respectively. Coin and Face images fed to BPNN without any pre-processing and with the dimensions of 128×128 and 100×100, respectively.

In Experiment 4.1 (Coin images), 16384-HN-2 architecture was used with 16 training and 64 testing patterns and, 5 and 10 hidden neurons could not be converged the training patterns. Similar results were obtained with other neuron numbers, however, the highest and lowest accuracy was obtained with 20 and 25 hidden neurons. Fluctuations occurred in generalization ability independent of hidden neuron number. Minimum and maximum iterations were recorded in 150 and 20 hidden neurons respectively with a change of 79.50 %.

In Experiment 4.2 (Face images), where 10000-HN-10 architecture, with 30 training and 70 test patterns was used, the minimum and maximum hidden neurons which were considered as 5 and 200 in this research, could not learn the training patterns. Lowest accuracy was obtained by the minimum hidden neuron number which could converge by 61.42 %. The highest accuracy was achieved with 50 hidden neuron number by 88.57 %. Minimum and maximum iterations were obtained by the highest and lowest hidden neuron numbers that could converge with a difference of 98.66 %.

Tab. VII shows the obtained results of Standard and Over-sized Processing Experiments for all hidden neuron numbers.

	Experiment 4.1		Experiment 4.2		Experiment 5.1	
Hidden Neuron No.	Iterations	Gen. Ability [%]	Iterations	Gen. Ability [%]	Iterations	Gen. Ability [%]
5	not converged		not converged		not converged	
10	not converged		60171	61.42	10709	22.91
15	1732	79.16	5804	85.71	7520	21.58
20	2717	**83.33**	9315	84.28	9527	23.00
25	1852	77.08	3172	84.28	**5194**	**23.58**
30	1675	81.25	2604	85.71	8567	21.91
50	1281	79.16	1739	**88.57**	not converged	
75	858	81.25	1165	78.57	not converged	
100	766	81.25	949	81.43	not converged	
150	**557**	79.16	**806**	75.71	not converged	
200	725	79.16	not converged		not converged	

Tab. VII *Results of high and over-sized processing experiments.*

4.5 Results of over-sized processing experiment

In OPEX, a single experiment with 983,040 TPE, was performed using Palm images. Images were resized to 64×64 and directly fed to NN without any preprocessing and 4096-HN-20 architecture was used with 240 training and 1200 testing patterns. Only hidden neuron numbers between 10 and 30 could reach defined RMS Error value. The highest accuracy and minimum iteration number was achieved with 25 hidden neurons by 23.58 % and 5194, respectively. The change between minimum and maximum iteration numbers was 51.49 %.

Tab. VII shows the obtained results of Over-sized Processing Experiments for all hidden neuron numbers.

5. Discussions and recommendations

In this section, obtained experimental results will be discussed in detail, and intervals for the hidden neurons of BPNN applications will be suggested.

5.1 Discussions on experimental results

In LPEX, it was observed that the use of over-sized hidden neurons either slightly decreased the generalization ability of BPNN by over-fitting the training patterns or produce similar results for any hidden neuron number. Analyses of learning efficiency of LPEX shows that increment of hidden neuron number in lower value of TPE requires more iterations to converge, however, the increment of TPE converts linear relationship between hidden neuron number and iterations.

In MPEX experiments, it was observed that the highest accuracy rates depend on the number of training patterns and the usage of a large number of training patterns requires the use of more hidden neuron numbers even TPE is smaller than other experiments. It should also be noticed that the number of training patterns are also affects the learning ability of NN. If the number of training patterns is small enough as in Experiments 2.1, 2.2, and 2.5, the iteration numbers decrease linearly according to the increment of hidden neuron number and, maximum and minimum number of iterations obtained with minimum and maximum hidden neuron numbers considered in these research. However, the increment of training patterns causes fluctuations of iterations during the convergence as in Experiments 2.3 and 2.4 which 200 and 280 training patterns are considered.

In SPEX, even the input neurons were low, the usage of high training patterns and more output neurons caused inefficient learning with 5 hidden neuron number during the training and not convergence of BPNN with oversized hidden neuron numbers. With more input neurons but less training patterns and output neurons, BPNN produced similar results for all hidden neuron numbers. Iterations decreased almost linearly with low training patterns and output neurons however, there was not a linear decrement in iteration numbers according to the hidden neuron increment for high training patterns and output neurons, and serious fluctuations occurred.

In HPEX, it was observed that even the training patters were low enough, over or undersized hidden neuron numbers may not learn or have difficulties during the learning of the training patterns. Also increment of output neurons which depend on the application, caused BPNN not to converge even with the highest hidden neuron number considered. Although little fluctuations were observed in 20 hidden neuron numbers, iterations were generally decreased linearly by the increment of hidden neuron numbers. However, it should be noticed that, maximum iteration in Experiment 4.1 is obtained by 20 hidden neurons which also achieved the highest recognition rate.

In OPEX, it was observed that the fluctuations occurred both in generalization ability and learning efficiency. Increment of TPE even with the same training patterns causes BPNN not to converge in over-sized hidden neurons.

When we analyze the learning efficiency of BPNN, the obtained results show that the increment of the hidden neuron numbers generally helps to decrease the number of iterations during the learning. However, the use of a large number of training patterns causes small or big fluctuations and unsteady iteration numbers even with a higher number of hidden neurons. It was observed that the increment of training patterns caused NN not to converge the training patterns in over-sized hidden neurons, and the increment of both TPE and training patterns caused NN not to converge both with over and under-sized hidden neuron numbers. It was not common for all experiments, but we can conclude that the lowest hidden neuron number generally needs more iterations to converge, as expected.

Difference between recorded maximum and minimum iteration numbers for the corresponding experiment varied from 8 % to 98 %, however convergence in minimum or maximum iterations does not mean that learning is more effective than others. Over-fitting training data occurred in many experiments and the analysis of the generalization ability of NN showed that the effect of the hidden neuron number was not linearly related to TPE, but related to the number of training patterns. In experiments, where the training patterns were higher than 60, it was observed that the highest accuracy needs higher numbers of hidden neurons that can converge.

5.2 Recommendations

As can be seen in the obtained results, there is not a linear relationship between hidden neuron numbers and the generalization ability of BPNN. Therefore, we can categorize the experiments according to TPE and training patterns as it was done in this research, to set a non-linear relation between optimum accuracy and hidden neuron numbers. Thus, researchers will be able to test the neural system between the recommended minimum and maximum hidden neuron numbers for their particular applications.

After analyses of fourteen experiments, it can be suggested that, if the training patterns are less than 60, 5 hidden neurons can be used for low, medium, and standard processing elements and, hidden neuron numbers between 20 and 50 should be considered in high-processing elements. For the applications those training patterns are more than 60, researchers should consider 5-10 hidden neurons for low, 75-200 hidden neurons for medium, 20-30 hidden neurons for standard and high processing elements. For over-sized processing elements, it is not recommended for any number of training patterns to use hidden neuron numbers higher than 30 and less than 10. Tab. VIII shows the suggested intervals for hidden neurons.

	TrP <60		TrP >60	
TPE	Min. HN	Max. HN	Min. HN	Max. HN
Low	5	–	5	10
Medium	5	–	75	200
Standard	5	–	20	30
High	20	50	20	30
Over-sized	10	30	10	30

Tab. VIII *Suggested hidden neuron intervals.*

6. Conclusions

Determination of optimum hidden neuron numbers in shallow neural networks is a big challenge and based on trial and error during performing the experiments. Low and high input-output neurons and, limited and increased training patterns should be considered in different experiments to analyze the effect of hidden neuron numbers. In this paper, fourteen varied experiments are performed in order to observe, analyze, and conclude the effect of the hidden neuron numbers on learning ability and generalization rates of neural networks. Experiments consist both numerical datasets and image-based databases and, divided into five categories according to the total processing elements of neural networks. Eleven different hidden neuron numbers which can be categorized as over, under, and normal-sized hidden neuron numbers according to the recent researches, are considered.

After performing fourteen experiments, it is once again proven that there is not a linear relationship between the number of hidden neurons and the generalization ability of BPNN which means an increment of hidden neuron number does not improve the generalization ability directly. However, performed experiments allowed us to determine hidden neuron intervals according to the total processing elements and the number of training patterns of the applications to minimize the time loss during trial and error for the determination of optimum hidden neuron number for particular application.

In addition to these analyses, it is observed that the effective data preparation step decreases the need for higher hidden neuron number and increases the recognition rates for specific application even needs higher iteration numbers. Varying or effective selection of training patterns may increase the accuracy of the neural system for particular applications, however, the effect of hidden neuron numbers will not differ in those applications.

Future work will include the analysis of hidden neuron number effect for big data in shallow neural networks thus; recommendation of universal intervals for hidden neuron number. Also the effect of hidden layer numbers in deep backpropagation neural networks to find a linear or non-linear relationships between considered hidden layers will be analyzed.

Acknowledgement

Portions of the research in this paper use the CASIA Palmprint Database collected by the Chinese Academy of Sciences' Institute of Automation (CASIA).

References

[1] LOPES N., RIBEIRO B. An evaluation of multiple feed-forward networks on GPUs. *International Journal of Neural Systems*. 2011, 21(1), pp. 31–47, doi: 10.1142/S0129065711002638.

[2] RUMELHART D.E., HINTON G.E., WILLIAMS R.J. Learning representations by back-propagating errors. *Nature*. 1986, 323, pp. 533–536, doi: 0.1038/323533a0.

[3] SINGH K. R., CHAUDHURY S. Efficient technique for rice grain classification using back-propagation neural network and wavelet decomposition. *IET Computer Vision*. 2016, 10(8) pp. 780–787, doi: 10.1049/iet-cvi.2015.0486.

[4] DIMILILER K., ZARROUK S. ICSPI: intelligent classification system of pest insects based on image processing and neural arbitration. *Applied Engineering in Agriculture*. 2017, 33, pp. 453–460, doi: 10.13031/aea.12161.

[5] RUBIO J. J. Modified optimal control with a backpropagation network for robotic arms. *IET Control Theory and Applications*. 2012, 6(14) pp. 2216–2225, doi: 10.1049/iet-cta.2011.0322.

[6] KUMAR J., SINGH A. K. Workload prediction in cloud using artificial neural network and adaptive differential evolution. *Future Generation Computer Systems*. 2018, 81, pp. 41–52, doi: 10.1016/j.future.2017.10.047.

[7] KHASHMAN A., SEKEROGLU B. Document image binarisation using a supervised neural network. *International Journal of Neural Systems*. 2008, 18(5), pp. 405–418, doi: 10.1142/S0129065708001671.

[8] DOUGHERTY G., *Pattern Recognition and Classification*. Germany: Springer. 2013, doi: 10.1007/978-1-4614-5323-9.

[9] ZENG X., YEUNG D. S. Hidden neuron pruning of multilayer perceptrons using a quantified sensitivity measure. *Neurocomputing*. 2016, 69(7-9), pp. 825–837, doi: 10.1016/j.neucom.2005.04.010.

[10] KWOK T. Y., YEUNG D. Y. Constructive algorithms for structure learning in feed forward neural networks for regression problems. *IEEE Transactions on Neural Networks*. 1997, 8(3), pp. 630–645, doi: 10.1109/72.572102.

[11] WYNEE-JONES M. Node splitting: a constructive algorithm for feedforward neural networks. *Neural Computing and Applications*, 1993, 1(1), pp. 17–22, doi: 10.1007/BF01411371.

[12] YEUNG D. S., ZENG X. Hidden neuron pruning for multilayer perceptrons using a sensitivity measure. In: *Proceedings of 1st International Conference on Machine Learning and Cybernetics*, 2002, pp. 1751–1757.

[13] ENGELBRECHT A. P. A new pruning heuristic based on variance analysis of sensitivity information. *IEEE Transcations on Neural Networks*, 2001, 12(6), pp. 1386–1399, doi: 10.1109/72.963775.

[14] FLETCHER L., KATKOVNIK V., STEFFENS F. E., ENGELBRECHT A. P. Optimizing the number of hidden nodes of a feedforward artficial neural network. In: *IEEE World Congress Computational Intelligence International Joint Conference on Neural Networks*, 1998, pp. 1608–1612.

[15] REED R. Pruning algorithms – a survey. *IEEE Transactions on Neural Networks*. 1993, 4(5), pp. 740–747, doi: 10.1109/72.248452.

[16] CASTELLANO G., FANELLI A. M., PELILLO M. An iterative pruning algorithm for feedforward neural networks. *IEEE Transactions on Neural Networks*. 1997, 8(3), pp. 519–531, doi: 10.1109/72.572092.

[17] SILVESTRE M. R., LING L. L. Pruning methods to MLP neural networks considering proportional apparent error rate for classification problems with unbalanced data. *Measurement*. 2014, 56, pp. 88–94, doi: 10.1016/j.measurement.2014.06.018.

[18] TONG Z., TANAKA G. A pruning method based on weight variation information for feedforward neural networks, *IFAC-PapersOnLine*, 2015, 48(18), pp. 221–226, doi: 10.1016/j.ifacol.2015.11.040.

[19] SHAFI I., AHMAD J., SHAH S. I., KASHIF F. M. Impact of varying neurons and hidden layers in neural network architecture for a time frequency application. In: *Proc. 2006 IEEE International Multitopic Conference*, 2007, pp. 188–193, doi: 10.1109/INMIC.2006.358160.

[20] LI Y., DONG W., YANG Q., ZHAO J., LIU L., FENG S. An automatic impedance matching method based on the feedforward-backpropagation neural network for WPT system. *IEEE Transactions on Industrial Electronics*, doi: 10.1109/TIE.2018.2835410.

[21] LIU H., GUO K., ZHANG Z., YU D. et al. High-power LED Photoelectrothermal analysis based on backpropagation artificial neural networks. *IEEE Transactions on Electron Devices*. 2017, 64(7), pp. 2867–2873, doi: 10.1109/TED.2017.2701346.

[22] PENG L., LIU J., SHENG M., ZHANG Y., HOU D., ZHENG Y., LI Y. 3D indoor localization based on spectral clustering and weighted backpropagation neural networks. *Proc. IEEE Int. Conference on Communications in China*, 2017, pp. 1–6, doi: 10.1109/ICCChina.2017.8330353.

[23] ANIFAH L., HARIMURTI R., RUSIMAMTO P. W., HARYANTO, PERMATASARI Z., MUHAMAD A. R., Cancer lungs detection on CT scan image using artificial neural network backpropagation based gray level co-occurence matrices feature. In: *Proc. 2017 International Conference on Advanced Computer Science and Information Systems ICACSIS*, 2017, pp. 327–331, doi: 10.1109/ICACSIS.2017.8355054.

[24] DIMILILER K., EVER Y. K., UGUR B. ILTDS: intelligent lung tumor detection system on CT images. *Advances in Intelligent Systems and Computing*. 2016, 530, pp. 225–235, doi: 10.1007/978-3-319-47952-1_17.

[25] YUAN J., YU S. Privacy preserving back-propagation neural network learning made practical with cloud computing. IEEE Transactions on Parallel and Distributed Systems. 2014, 25(1), pp. 212–221, doi: 10.1109/TPDS.2013.18.

[26] WU K., ZHONG Y., WANG X., SUN W. A novel approach to subpixel land-cover change detection based on a supervised back-propagation neural network for remotely sensed images with different resolutions. *IEEE Geoscience and Remote Sensing Letters*. 2017, 14(10), pp. 1750–1754, doi: 10.1109/LGRS.2017.2733558.

[27] WEI P., CHENG C., LIU T. A photonic transducer-based optical current sensor using back-propagation neural network. *IEEE Photonics Tech. Letters*. 2016, 28(14), pp. 1513–1516, doi: 10.1109/LPT.2016.2557339.

[28] MOSHKBAR-BAKHSHAYESH K., GHOFRANI M. B. Development of efficient identifier for nuclear power plant transients based on latest advances of error back-propagation learning algorithm. *IEEE Transactions on Nuclear Science*. 2014, 61(1), pp. 602–610, doi: 10.1109/TNS.2013.2292898.

[29] KHASHMAN A., SEKEROGLU B., DIMILILER B. ICIS: a novel coin identification system. *Lecture Notes in Control and Information Sciences*. 2006, 345, pp. 913–918, doi: 10.1007/11816515_113.

[30] KHASHMAN A., DIMILILER B. Medical radiographs compression using neural networks and haar wavelet. In: *Proc. of the IEEE International Conference on 'Computer as a Tool'*, 2009, pp. 1448–1453, doi: 10.1109/EURCON.2009.5167831.

[31] ADALI T., SEKEROGLU B. Analysis of micrornas by neural network for early detection of cancer. *Procedia Technology*. 2012, 1, pp. 449–452, doi: 10.1016/j.protcy.2012.02.100.

[32] DIMILILER K. IBFDS: intelligent bone fracture detection system. *Procedia Computer Science*. 2017, 120, pp. 260–267, doi: 10.1016/j.procs.2017.11.237.

[33] FISHER R. A. The use of multiple measurements in taxonomic problems. *Annual Eugenics*. 1936, 7(2), pp. 179–188, doi: 10.1111/j.1469-1809.1936.tb02137.x.

[34] FORINA M., LEARDI R., ARMANINO C., LANTERI S. PARVUS – an extendible package for data exploration, classification and correlation, Institute of Pharmaceutical and Food Analysis and Technologies, 1988, Via Brigata Salerno, 16147 Genoa, Italy.

[35] DIACONIS P., EFRON B. Computer-intensive methods in statistics. *Scientific American*. 1983, p. 248, doi: 10.1038/scientificamerican0583-116.

[36] CESTNIK G., KONENENKO I., BRATKO I. Assistant-86: a knowledge-elicitation tool for sophisticated users. In: *Bratko,I., Lavrac, N. (Ed.), Progress in Machine Learning*. 1987, pp. 31–45.

[37] GORMAN R. P., SEJNOWSKI T. J. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*. 1988, 1(1), pp. 75–89, doi: 10.1016/0893-6080(88)90023-8.

[38] TABTAH F. Autism spectrum disorder screening: machine learning adaptation and DSM-5 fulfillment. In: *Proc. 1st International Conf. on Medical and Health Informatics*, 2017, pp. 1–6, doi: 10.1145/3107514.3107515.

[39] CAMPOS A., BERNARDES J., GARRIDO A., MARQUES-DE-SA J., PEREIRA-LEITE L. SisPorto 2.0: a program for automated analysis of cardiotocograms. *Journal of Matern Fetal Med*. 2010, 9(5), pp. 311–318, doi: 10.3109/14767050009053454.

[40] The Database of Faces. AT&T Laboratories Cambridge, http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[41] CASIA Multi-Spectral Palmprint Database, http://biometrics.idealtest.org/.

[42] SEKEROGLU B. Classification of sonar images using back propagation neural network. In: *Proc. IEEE Geoscience and Remote Sensing Society Symposium*, 2004, pp. 3092–3095.