Multilayer Neural Network Training Error when AMSGrad, Adam, AdamMax Methods Used

Bohdan Melnyk, Serhiy Sveleba, Ivan Katerynchuk, Ivan Kuno and Volodymyr Franiv

Ivan Franko National University of Lviv, 1, Universytetska St., Lviv, 79000, Ukraine

Abstract

The multilayer neural network training errors when the optimization methods of learning Adam, AdamMax, AMSGrad used are considered. The multilayer neural network used for the recognition of printed numbers. It was established that with an increase of the learning speed value there are mode of underlearning, satisfactory learning, and a chaotic learning mode. The process of neuron retraining is characterized by the appearance of local minima of the error function. The chaotic mode of learning is described by the process of doubling the number of existing local minima. The work defines and describes the mechanism for determining the optimal learning speed, which corresponds to the appearance of the first harmonic of the error function, or the learning speed at which the first loss of stability of the learning system is observed on the branching diagram. The conducted studies of the learning error of multilayer neural network when using the optimization learning methods AMSGrad, Adam, AdamMax prove that these methods do not affect the value of the optimal learning speed, and it does not depend on the change of the optimization parameter β_2. For all considered optimization methods, it is 0.45.

Keywords

The multilayer neural network, the training error, optimization methods

1. Introduction

An important aspect in the process of learning and testing neural networks is to avoid the process of their retraining. This is a key task in the development of machine learning models. The most famous ways to avoid retraining:

- ***using a validation set*** [1]. The data is divided into training, validation and test sets. The validation set is used to evaluate the performance of the model during training, which allows timely detection of overtraining and taking measures to avoid it.
- ***use of regularization methods*** [2], such as L1 or L2 regularization, which add corrections to the magnitude of model parameters. This helps to avoid over-complexity of the model.
- ***reducing the number of parameters***, namely using fewer layers or neurons to avoid overtraining [3]. This is especially important when computing resources are limited.
- ***application of cutting methods(dropout)*** [4] during training, when certain neurons are randomly dropped during each iteration. This prevents the model from adapting to specific noise dependencies in the training data.
- ***cross check*** [5], that is, instead of a one-time division of the data set into training and validation, cross-validation is used, which allows for a more accurate assessment of the overall performance of the model.

COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine

bohdan.melnyk@lnu.edu.ua (B. Melnyk); serhiy.sveleba@lnu.edu.ua (S. Sveleba); ivan.katerynchuk@lnu.edu.ua (I. Katerynchuk); ivan.kuno@lnu.edu.ua (I. Kuno); volodymyr.franiv@lnu.edu.ua (V. Franiv)

0000-0001-6399-6317 (B. Melnyk); 0000-0002-0823-910X (S. Sveleba); 0000-0001-8877-8324 (I. Katerynchuk); 0000-0001-6092-7949 (I. Kuno); 0000-0001-9856-1962 (V. Franiv)

- ***using an early stop (early stopping)*** [6], that is, the learning process is interrupted when the performance of the model on the validation set begins to deteriorate. This may indicate the beginning of retraining.
- ***using deep learning architectures***, which in themselves are less prone to retraining [7]. Some architectures, such as neural networks with multiple layers of abstraction, may be less prone to overtraining.
- ***combining these methods*** to get the best results and avoid overtraining the neural network.

All of the above methods are aimed at avoiding retraining, but not at the very cause of its appearance. According to the results of work [8], the process of retraining neural networks is associated with the appearance of local minima on the target error function. Thus, it was noted in [8] that when the error function approaches the global minimum, the appearance of local minima increases. Based on the analysis of the objective function of the learning error with the help of a logistic function describing the doubling process, it was noted in the paper [9] that the appearance of local minima is primarily caused by the retraining of neurons. When approaching the global minimum, due to the inhomogeneity of the learning process, the number of retrained neurons increases, and in the first approximation this process can be described as a frequency doubling process. This is particularly indicated by the Fourier spectra of the target error function and the obtained branching diagrams [9]. Also, as it was shown in [9], the process of relearning is related to the choice of the input array and its heterogeneity. Based on the maps of dynamic modes, according to [9], the heterogeneity of the input array serves as a catalyst for neuron retraining. It was also stated in [9] that the following learning modes are inherent in the neural network learning process: undertraining, satisfactory learning mode, and retraining mode. According to the branching diagrams obtained from the objective function of the error, the retraining mode of the neural network is characterized by both partial retraining and chaotic training. According to the Fourier spectra of the learning error function, the transition to the chaotic learning regime is accompanied by a doubling of the number of local minima. Along with this, transparency windows are observed on the branching diagrams, which testify to the emergence of a satisfactory learning process of the neural network. Based on the above, in [9] a model was proposed to describe the appearance of the relearning mode for a multilayer neural network with inverse error propagation. Since in neural networks, input values are processed on each neuron from all neurons of the previous layer (on the second layer from all input values), the existence of a significant number of periodicities will be inherent in the error function. This behavior of the error function is caused by the retraining of individual neurons. That is, the increase in the number of local minima of a multilayer neural network when approaching the global minimum is due to the process of retraining a certain number of neurons. Such retraining causes the periodic behavior of the error function to appear. Since the error function of the neural network is a symbiosis of the error functions of each neuron, its behavior will be characterized by a spectrum of possible oscillation frequencies. Depending on such a parameter as the learning rate (*alpha*), all learning modes will be inherent in the neural network, including the chaotic learning mode. In the retraining mode of a small number of neurons, the error function is a periodic function and is described by several different oscillations. Under this condition, the learning error function is characterized by the existence of several local minima. As a result of doubling the number of local minima, the neural network goes into a chaotic learning mode. In this learning mode, the error function of the neural network is described by a set of existing oscillations, and the average wave vector over such an ensemble of oscillations can take an incommensurate value to the existing oscillations. Therefore, the emerging chaotic learning mode of the neural network is characterized by the retraining of a significant number of neurons, and their number changes dramatically when the learning speed changes.

This work is devoted to the consideration of the algorithm for avoiding the retraining mode of the neural network, and the comparative analysis of the learning and testing error function under the condition of practically no retraining mode of the neural network, when using the

most effective optimization methods of training, such as Adam, AdamMax and AMSGrad. Since the process of relearning is strongly influenced by both the sample and the array of setting the numbers themselves, we will conduct this analysis by considering the influence of both the sample and the array of setting the number itself for both homogeneous and heterogeneous input arrays.

2. Methodology

Printed digit representation arrays such as TensorFlow and Keras, which have the MNIST dataset, which also includes handwritten digits, are used to train a digit recognition model. Since an array of 28x28 pixels in gradation of shades of gray is mainly used for displaying numbers, these arrays for displaying numbers can be classified as homogeneous arrays. The conducted preliminary studies of maps of dynamic modes show that these arrays can be considered as homogeneous. The analysis for these arrays will be given in our next work. We will focus at the analysis of the learning process of the neural network with the help of printed numbers. Due to their simple form of representation of digital data, as a set of zeros and ones, they are widely used for machine learning and pattern recognition. The paper deals with the array of 3x5 and 4x7 numbers. An increase in the number representation array will contribute to the movement towards the uniformity of the array. The sample will consist of five options for presenting the number. That is, one version without distortion of the numbers and four distorted with an error of 15-20% (for the 3x5 array) and 11-15% (for the 4x7 array). In this representation of the number, one unit is replaced by a zero or vice versa. In the first approximation, we will consider such an array to be homogeneous. A non-homogeneous array will be considered an array in which the number representation error is close to 50%. For this, two variants of the number representation with an error of ≈ 50% were added to the number array display sample. That is, such an array, which was given by a set of zeros and ones, did not correspond to any of the numbers. Calculations were performed in the Python programming environment for a neural network with three hidden layers of 15 and 28 neurons in each layer, respectively, for a 3x5 and 4x7 digit representation array.

The learning error function was analyzed using a logistic function of the following form:

$$x_{n+1} = alpha - x_n - x_n^2$$

where n is a step, *alpha* is a parameter that determines the learning rate. The selection of β_1 and β_2 parameter values for the considered optimization methods Adam, AdamMax and AMSGrad was carried out according to work. According to [9], the sigmoidal function played the role of the activation function. The architecture of the multilayer neural network was chosen according to [9], where it was noted that this architecture has the lowest learning error. Testing was carried out when presenting a number with an error of 15-20% (for a 3x5 array) and 11-15% (for a 4x7 array).

3. The Adam method

3.1. Homogeneous array, representation of number in a 3x5 array

Adam Method is quite popular in deep learning because it works effectively with different types of neural network architectures and different tasks. This method is an optimization algorithm used for neural network training, in particular in deep learning. It combines the ideas of gradient descent optimization methods (for example, the method of moments and RMSProp) with additional corrections.

The main idea of the Adam method is to use the exponentially weighted mean gradient (from RMSProp) for each parameter, and also use the exponentially weighted mean square of the

gradient (similar to the method of moments). This allows the algorithm to effectively adapt to different magnitudes and directions of gradients.

This method is described by the following formulas:

This is the calculation of the exponentially weighted mean gradient: $m_n=\beta_1 m_{n-1}+(1-\beta_1)g_n$

This is the calculation of the exponentially weighted mean square of the gradient: $v_n=\beta_2 v_{n-1}+(1-\beta_2)g_n^2$

Offset correction: $\widehat{m_n} = m_n/(1-\beta_1^n), : \widehat{v_n} = v_n/(1-\beta_2^n)$

Weights are updated according to the formula: $w_{n+1} = w_n - \eta\widehat{m_n}/\sqrt{\widehat{v_n}+\epsilon}$

In these formulas:

g_n is the gradient of the target error function at point *n*, where *n* is the number of iterations,

m_n is exponentially weighted average of the gradient,

v_n is exponentially weighted average of the square of the gradient,

β_1 and β_2 - parameters, usually close to 1, which control the degree of attenuation of the previous values, according to the work [10], respectively, 0.9 and 0.99, 0.999 or 0.9999 are chosen

η - learning rate (learning rate),

w_n- parameters of the model at *n*,

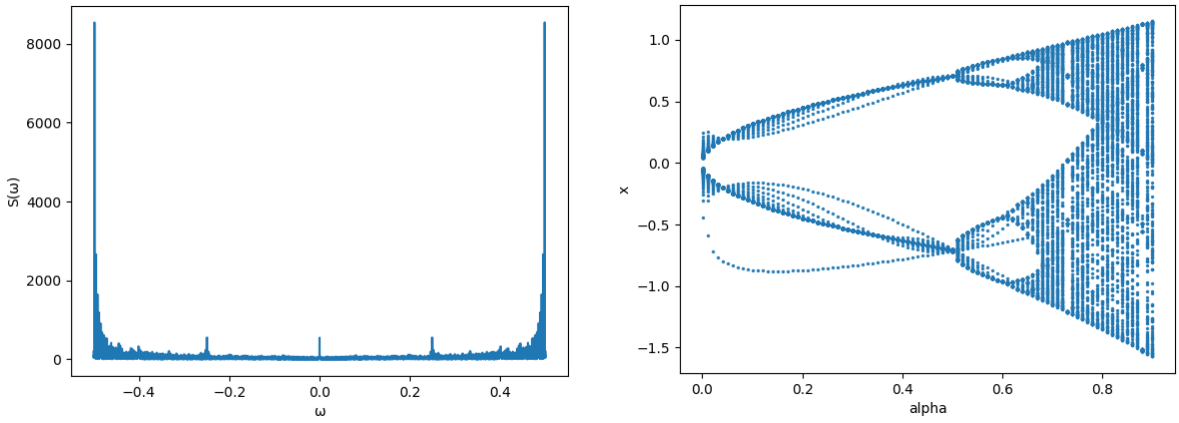
ϵ - a small addition for numerical stability (usually a very small number, for example, 10^{-8}).

Fig. 1 shows the Fourier spectra at the learning speed that corresponds to the retraining of the neural network (Fig. 1, a, *alpha*=0.9) and the branching diagram (Fig. 1, b) at *N*=100 iterations. According to Fig. 1, the Fourier spectrum is characterized by a wide range of existing harmonics, which testify to the existence of the retraining mode of the neural network. According to the branching diagram shown in Fig. 1b, already at the initial stage of neural network training, the learning error function on each neuron is a complex functional dependence. Starting with a learning speed of *alpha*>0.45, the process of the appearance of local minima should be followed. According to the branching diagram, their number (and therefore the number of neurons that are inherent in the relearning process) doubles when the learning speed increases. That is, the further increase in the learning speed begins to be described by the process of doubling the number of local minima in the learning error function. In the final case, this leads to the appearance of chaotic behavior of the learning error function. Considering in combination with Fourier spectra and branching diagrams, the conclusion is suggested that the increase in the number of harmonics on the Fourier spectra and the number of branches on the branching diagram is associated with the appearance of local minima in the behavior of the target error function. And the appearance of local minima is due to the retraining of neurons.

Therefore, a chaotic learning mode occurs in a multilayer neural network, which is caused by the appearance of local minima, which arise as a result of an increase in the learning speed, during which the process of relearning is traced. The resulting chaotic learning mode is sensitive to changes in the parameters of the multilayer neural network. A slight change in the multilayer neural network parameters causes significant changes in the multilayer neural network training mode. The appearance of local solutions (local minima) as a result of an increase in the speed of learning leads to the appearance of bifurcations on the dependence of the learning error on the number of epochs and all this associated with the process of relearning neurons.

One of the ways to solve this problem (avoid the chaotic learning regime of multilayer neural network) is to determine the parameters of the appearance of the harmonics of the objective function of the learning error, and on the branching diagram, the presence of a bifurcation. Other words: the determination of the value of the learning speed, at which the process of the appearance of local minima due to the relearning process of neurons takes place. This mechanism assumes the absence of local minima, and therefore the retraining regime. The algorithm for solving this problem consists in determining the optimal value of the learning speed, and therefore the optimal value of the learning error at which the first harmonic occurs on the learning error function.

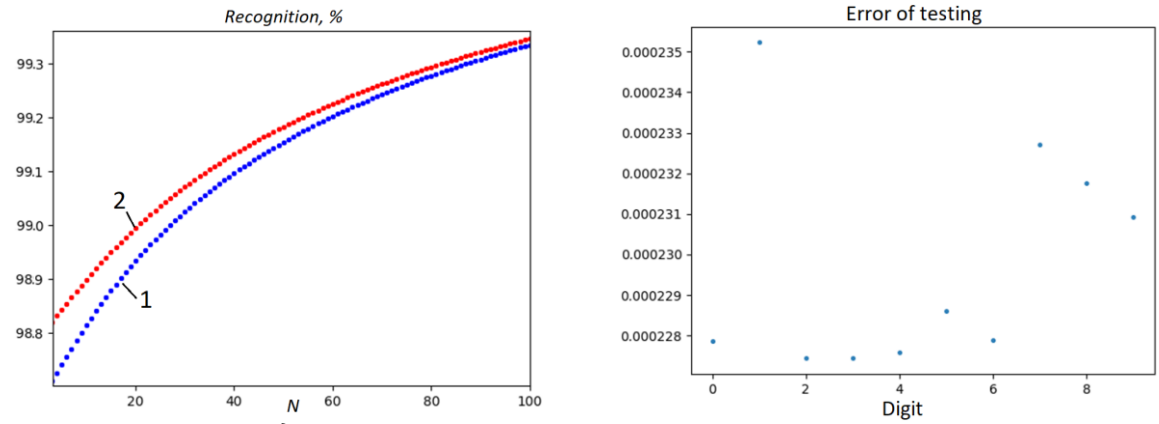
Under these conditions, the obtained optimal value of the learning speed (*alpha*) is equal to 0.4501 for each digit. This shows that the Adam method effectively selects the learning parameters for different input data. The learning error in this case varied from 2.608e-05 to 2.6295e-05, which is a negligible value. Such results indicate the high accuracy of neural network learning and its ability to effectively implement the learning process.



a) *b*)

Figure **1**: Fourier spectra *a*), and branch diagram for the function of the learning error from the number of iterations *b*), subject to the use of the Adam optimization method, for a homogeneous array of dimensions 3x5, with β_1=0.9, β_2=0.999 and *N*=100 iterations.

Figure 2a shows the dependence of the quality of training (1) and testing (2) under the conditionβ_1=0.9, β_2=0.999 and *N*=100. The test curve shows a better result than the learning curve. Values of testing errors for different numbers are shown in Fig. 2, b. The test error for the digits "1", "7", "8" and "9" shows a worse result than for the other digits.



a) *b*)

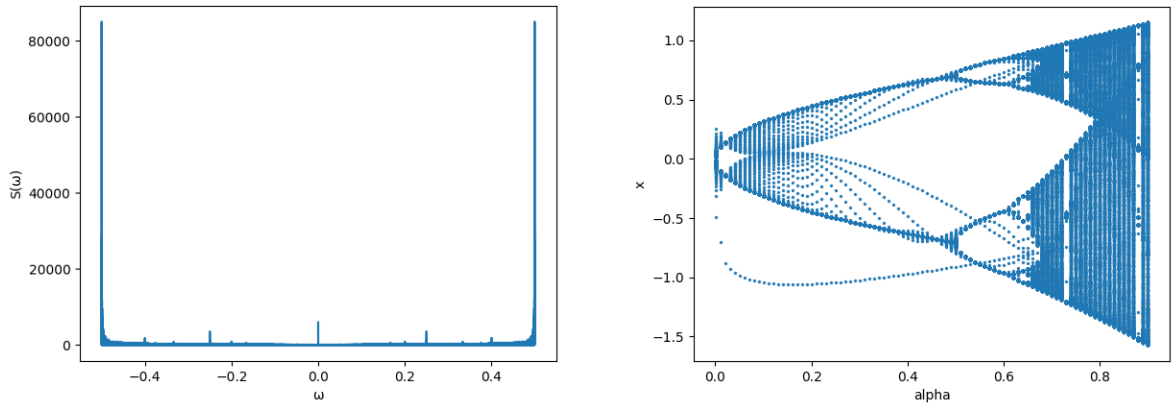
Figure **2**: Dependence of the value of digit recognition for the training array (1) and the test array (2) on the number of iterations, and the testing error for each digit, subject to the application of the Adam optimization method, for a homogeneous array of dimensions 3x5, with β_1=0.9, β_2=0.999 and *N*=100.

At 1000 iterations, a significant improvement in the accuracy and efficiency of the training model can be seen. The training error values, which range from 8.129e-06 to 8.139e-06 for different figures, are significantly lower compared to the previous data obtained at 100 iterations. According to the Fourier spectra and branching diagrams shown in Fig. 3, under the condition β_1=0.9, β_2=0.999 and 1000 iterations, the optimal learning rate was 0.4501.

When studying the influence of the optimization parameter β_2 the following results were obtained:

β_2=0.99, optimal alpha = 0.4501; learning error = 8.129e-06÷8.139e-06;
β_2=0.999, optimal alpha = 0.4501; learning error = 8.129e-06÷8.139e-06;
β_2=0.9999, optimal alpha = 0.4501; learning error = 8.129e-06÷8.139e-06.

That is, changing the optimization parameter β_2 within the accuracy of the experiment, it does not affect either the value of the optimal learning speed or learning errors.

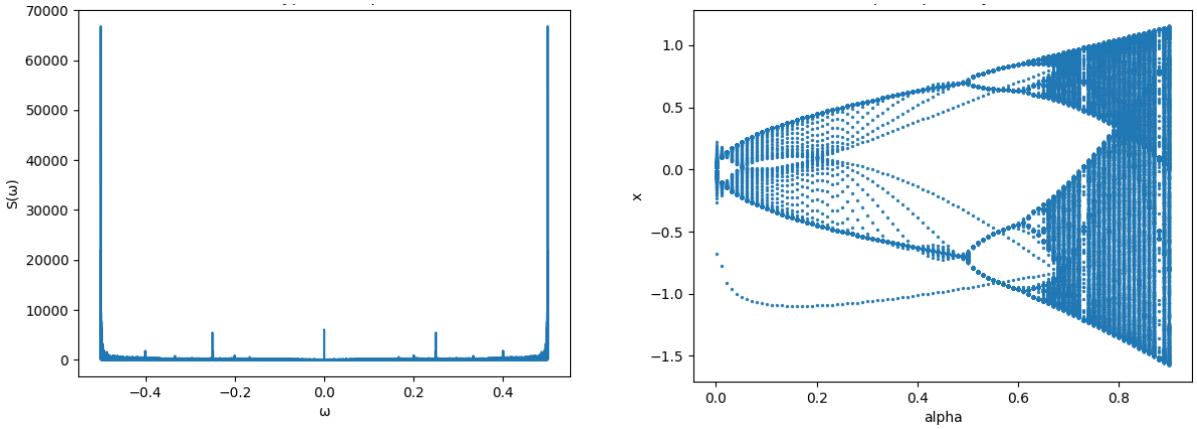


a) b)

Figure **3**: Fourier spectra *a*), and branch diagram for the function of the learning error from the number of iterations *b*), provided the optimization method is Adam, for a homogeneous array of dimensions 3x5, at β_2=0.999 and *N*=1000 iterations.

3.2. Heterogeneous array, representation of numbers in a 3x5 array

Figure 4 shows the Fourier spectra of the error function and the branching diagram of the learning rate. The resulting Fourier spectra and branching diagrams are practically identical to the Fourier spectra and branching diagrams for a homogeneous array.



a) *b*)

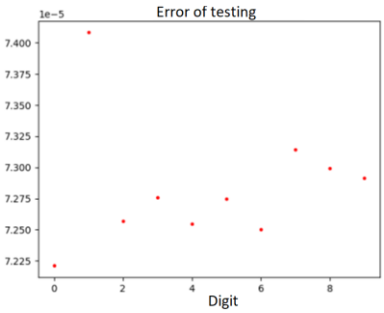
Figure **4**: Fourier spectra *a*), and branch diagram for the function of the learning error from the number of iterations *b*), provided the Adam optimization method, for a non-homogeneous 3x5 array, with β_2=0.99 and *N*=1000 iterations.

When studying the influence of the optimization parameter β_2 the following results were obtained:

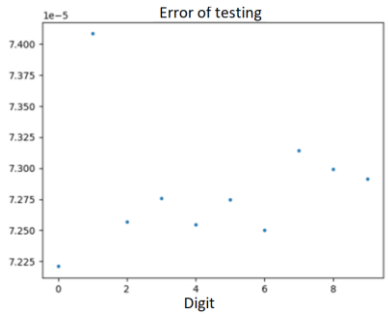
β_2=0.99, optimal *alpha* = 0.4501; learning error = 6.429e-06÷6.434e-06;
β_2=0.999, optimal *alpha* = 0.4501; learning error = 6.429e-06÷6.434e-06;
β_2=0.9999, optimal *alpha* = 0.4501; learning error = 6.429e-06÷6.434e-06.

That is, changing the optimization parameter β_2 within the accuracy of the experiment does not affect either the value of the optimal speed of learning or the accuracy of learning for a heterogeneous array. So, the change of the β_2 parameter for a homogeneous or non-homogeneous input array does not affect the training result when using the Adam optimization method.

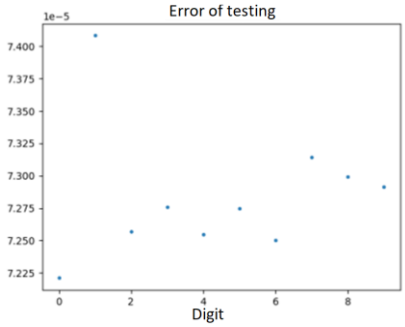
Consider how the homogeneity or non-homogeneity of the input array affects the testing error. Fig. 5 shows the testing error for different numbers when the optimization parameter β_2 is changed and constant value β_1=0.9, for homogeneous and heterogeneous input array. Within the accuracy of the experiment, the change in the value of the optimization parameter β_2 practically does not affect the testing error, both for a homogeneous array and for a non-homogeneous array. For the testing process, it does not depend on the value of β_2 there is a pattern that the testing error for the digits "1", "7", "8" and "9" shows a worse result than for other digits.



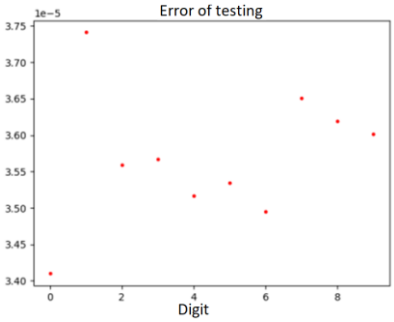
3x5 homogeneous array, at β_2=0.99.



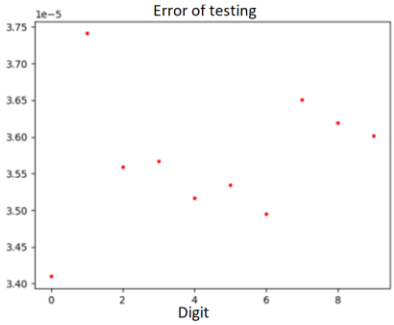
3x5 homogeneous array, at β_2=0.999.



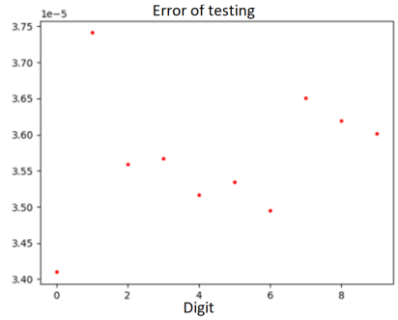
3x5 homogeneous array, at β_2=0.9999.



3x5 non-homogeneous array, at β_2=0.99.



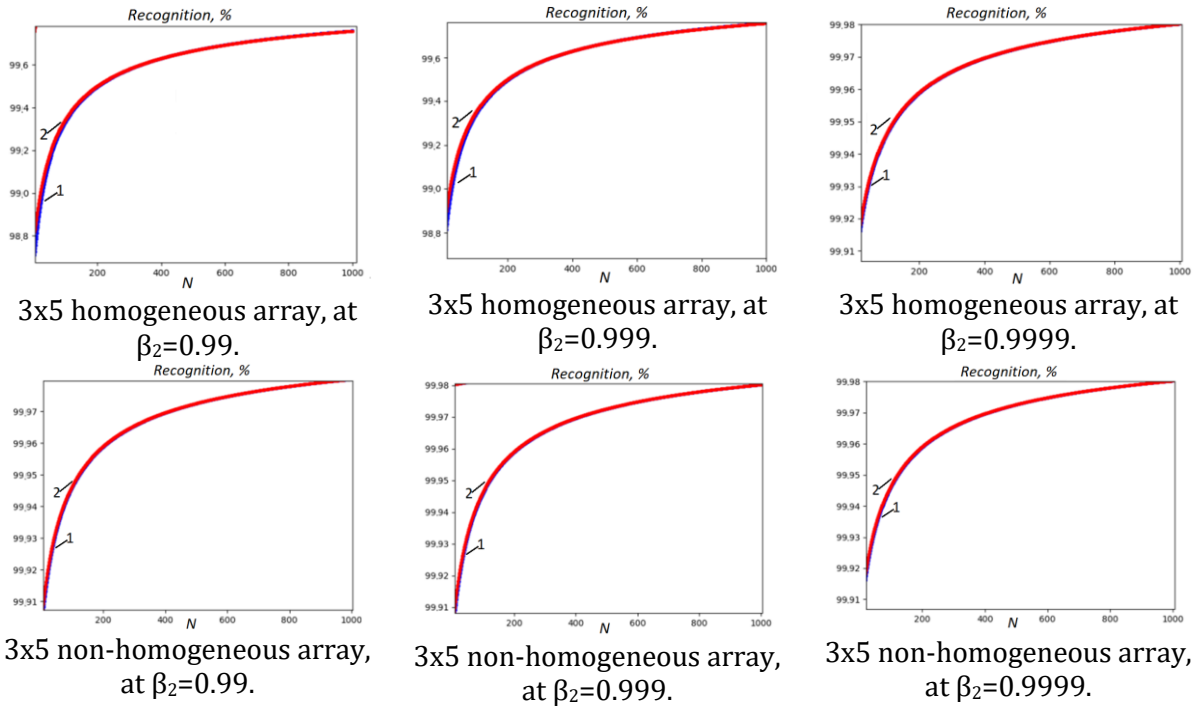
3x5 non-homogeneous array, at β_2=0.999.



3x5 non-homogeneous array, at β_2=0.9999.

Figure **5**: Testing error for different numbers when changing the optimization parameter β_2 and a constant value of β_1=0.9, for a homogeneous and heterogeneous input array, provided the Adam optimization method is used, *N*=1000 iterations.

Comparing the testing error for a homogeneous and non-homogeneous array, according to Fig. 5, a pattern can be observed that the testing error for a non-homogeneous array is almost two times smaller. If you compare the testing error with the learning error, the latter is almost an order of magnitude smaller. Although the dependences of training quality (1) and testing quality (2) shown in Fig. 6 do not reflect this. But comparing the dependencies obtained for a homogeneous input array and a non-homogeneous one, it should be noted that the steepness of the change in the quality of testing as well as training for a homogeneous array is greater and does not depend on the parameter β_2.



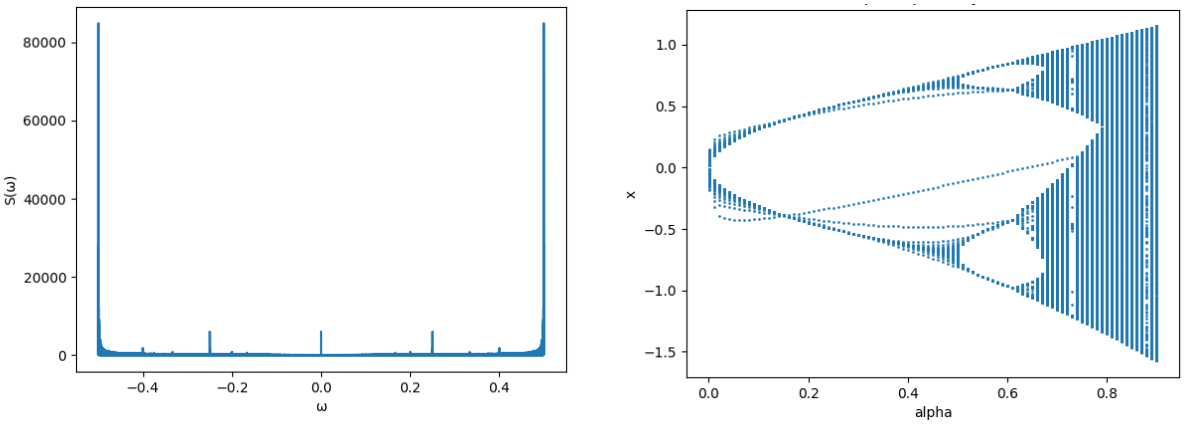
3x5 homogeneous array, at β_2=0.99. | 3x5 homogeneous array, at β_2=0.999. | 3x5 homogeneous array, at β_2=0.9999.

3x5 non-homogeneous array, at β_2=0.99. | 3x5 non-homogeneous array, at β_2=0.999. | 3x5 non-homogeneous array, at β_2=0.9999.

Figure **6**: The quality of training (1) and testing (2) from the number of iterations (N=1000) and the optimization parameter β_2 for a homogeneous and non-homogeneous array.

3.3. Homogeneous array, representation of numbers in a 4x7 array

Fig. 7 shows the Fourier spectrum and the branching diagram for the learning error function from the learning speed, subject to the application of the Adam optimization method, for a homogeneous array of dimensions 4x7, with β_2=0.999. The obtained Fourier spectra for both the 3x5 and 4x7 number arrays are characterized by the existence of harmonics, and the first, second and third are clearly visible in the spectra. The branching diagram shown in Fig. 7 is, in the first approximation, similar to the diagram for the 3x5 array (Fig. 4), with the only difference that the learning process in the interval *alpha*<0.4 is more uniform. That is, the correction of the magnitude of the weights for all neurons is almost the same.



a) *b*)

Figure **7**: Fourier spectra a) and branch diagram for the function of the learning error from the number of iterations b), subject to the application of the Adam optimization method, for a homogeneous array of dimensions 4x7, with β_2=0.999 and N=1000 iterations.

Consider the influence of the optimization parameter β_2 on the learning error at the optimal value of the learning speed for a homogeneous array of dimensions 4x7:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 5.955e-06÷5.96e-06;

β_2=0.999, optimal *alpha* = 0.4501; learning error = 5.955e-06÷5.96e-06;

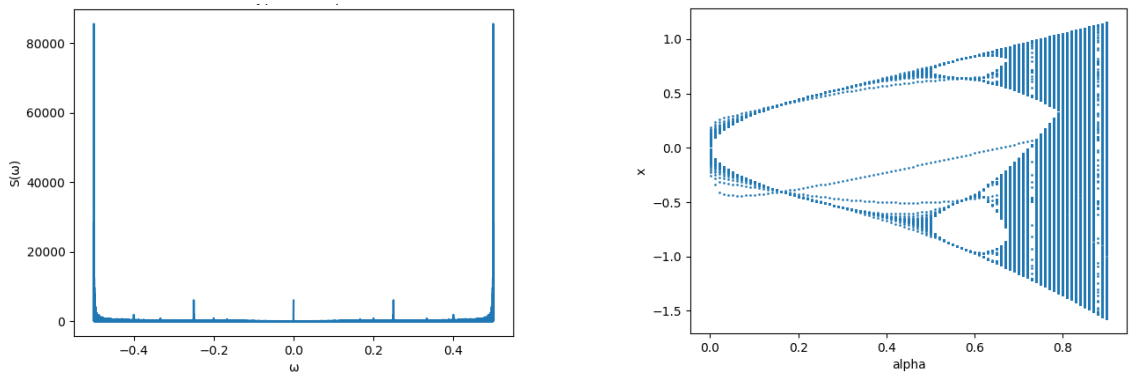
β_2=0.9999, optimal *alpha* = 0.4501; learning error = 5.955e-06÷5.96e-06.

Changing the optimization parameter β_2 within the accuracy of the experiment does not affect either the value of the optimal learning speed or the learning accuracy for a homogeneous array.

Increasing the number display array from 3x5 to 4x7 caused a decrease in the learning error.

3.4. Non-homogeneous array, representation of numbers in a 4x7 array

Fig. 8 shows the Fourier spectra and branching diagram obtained using the logistic function, for the learning error function under the condition of using the Adam optimization method, for a non-homogeneous 4x7 array, with β_2=0.999 and at 1000 iterations. The resulting Fourier spectra and branching diagram are similar to those for a homogeneous array. This shows that increasing the size of the number representation has a positive effect on the learning process of the neural network.



a) *b*)

Figure **8**: Fourier spectra a) and branching diagram for the function of the learning error from the number of iterations b), subject to the use of the Adam optimization method, for a non-homogeneous array with the size of the representation of numbers 4x7, with β_2=0.999 and *N*=1000 iterations.

Consider the influence of the optimization parameter β_2 on the learning error at the optimal value of the learning speed for a non-homogeneous array the size of the representation of the number is 4x7:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 4.709e-06÷4.714e-06;

β_2=0.999, optimal *alpha* = 0.4501; learning error = 4.709e-06÷4.714e-06;

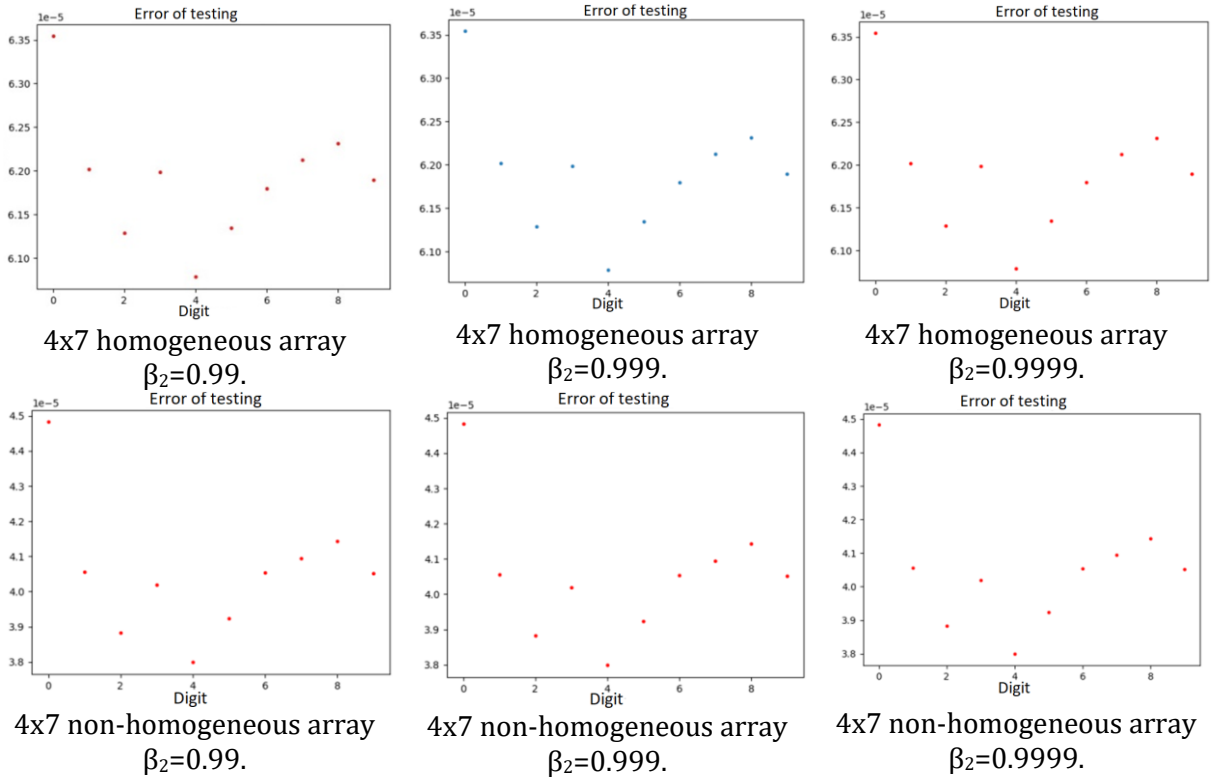
β_2=0.9999, optimal *alpha* = 0.4501; learning error = 4.709e-06÷4.714e-06.

Changing the optimization parameter β_2 does not affect either the value of the optimal learning speed or the learning error for a non-homogeneous array.

Increasing the dimension of array from 3x5 to 4x7 caused a decrease in the learning error not only for the homogeneous array but also for the non-homogeneous array.

Let's now consider how the homogeneity and non-homogeneity of the input array affects the testing error for the 4x7 digit display array. Fig. 9 shows the testing error for different numbers when the optimization parameter is changed β_2 and constant value β_1=0.9, for homogeneous and heterogeneous input array. Within the accuracy of the experiment, the change in the value of the optimization parameter β_2 practically does not affect the testing error, both for a homogeneous array and for a non-homogeneous array. For the testing process, it does not depend on the value of β_2 there is a pattern that the testing error for the digits "0" shows a worse result than for other digits.

Comparing the testing error for a homogeneous and non-homogeneous array, according to Fig. 9, a pattern can be observed that the testing error for a non-homogeneous array is almost a third smaller. Comparing the testing error with the learning error, the latter is almost an order of magnitude smaller.



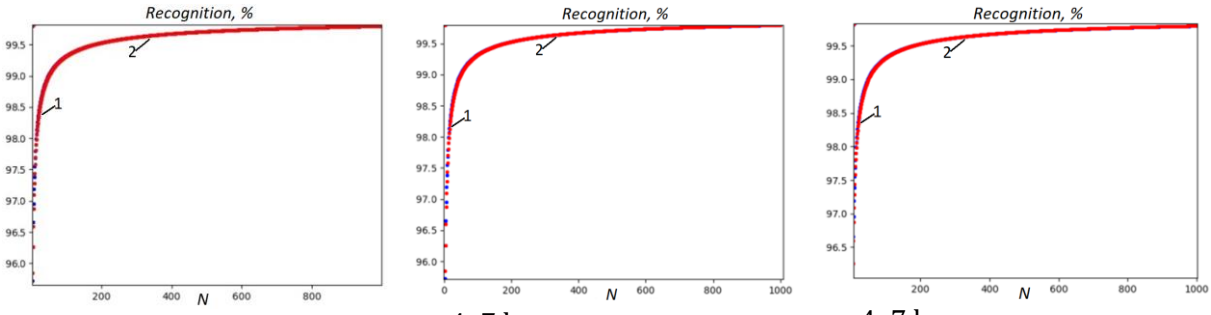
4x7 homogeneous array β_2=0.99. | 4x7 homogeneous array β_2=0.999. | 4x7 homogeneous array β_2=0.9999.

4x7 non-homogeneous array β_2=0.99. | 4x7 non-homogeneous array β_2=0.999. | 4x7 non-homogeneous array β_2=0.9999.

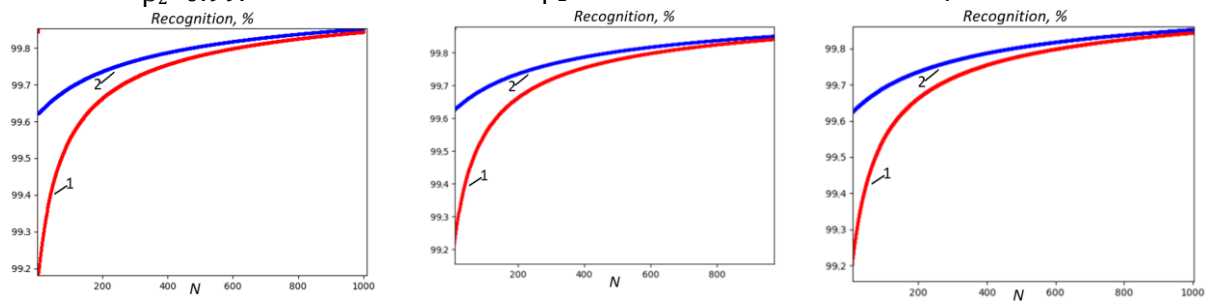
Figure **9**: Testing error for different numbers when changing the optimization parameter β_2 and constant value β_1=0.9, for homogeneous and non-homogeneous input array, *N*=1000 iterations.

The dependencies of training quality (1) and testing quality (2) shown in Fig. 10 do not reflect this. Comparing the obtained dependencies for a homogeneous input array, it should be noted that they practically coincide. Their coincidence is followed even when the optimization parameter is changed β_2. The interval of a sharp change in their dependence is followed up to 100 iterations. For a non-homogeneous array, as mentioned above, the training error is smaller than the testing error. Therefore, the learning quality curve is higher than the corresponding test curve. But in comparison with a homogeneous array of numbers, with a non-homogeneous array, the quality of training and testing is characterized by a smaller change in the number of iterations. This leads to the need to carry out the training and testing process with a greater number of iterations.

But an increase in the number of iterations when using a non-homogeneous array is accompanied by a smaller value of both training and testing errors. With a homogeneous input array, the learning speed is higher at the beginning, which then sharply slows down starting with 100 iterations.



4x7 homogeneous array β_2=0.99. | 4x7 homogeneous array β_2=0.999. | 4x7 homogeneous array β_2=0.9999.



4x7 non-homogeneous array β_2=0.99. | 4x7 non-homogeneous array β_2=0.999. | 4x7 non-homogeneous array β_2=0.9999.

Figure **10**: The quality of training (1) and testing (2) from the number of iterations and the optimization parameter β_2 for a homogeneous and non-homogeneous array.

4. Method AdamMax

4.1. Homogeneous array, representation of numbers in a 3x5 array

The AdamMax optimization method is an improved version of the Adam optimization algorithm. AdamMax can be useful for training deep neural networks, especially in cases where problems with instability or large gradients may arise. The main idea of AdamMax is to use the maximum norm for regularization [10]. AdamMax uses the maximum value of the absolute values of the gradients for each parameter instead of using the average of the squares of the gradients (as in Adam). This allows you to focus on large gradients, which can improve the stability and learning speed of the model.

The general AdamMax optimization algorithm consists in initializing the following values: learning rate, exponential average moments of the first and second orders, β_1 and β_2, the maximum norm (max_norm) used to limit the gradients.

This method is given by the following formulas:

Calculation of the exponentially weighted mean gradient: $m_n=\beta_1 m_{n-1}+(1-\beta_1)g_n$

Calculation of the exponentially weighted mean square of the gradient: $v_n=\beta_2 v_{n-1}+(1-\beta_2)g_n^2$

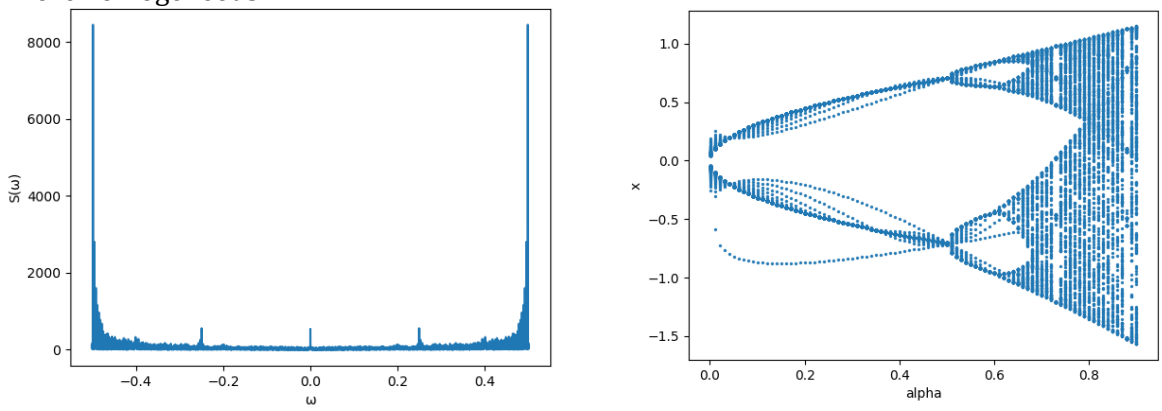
$v_n=\max(\beta_2 \cdot v_{n-1}, |g|)$,

Offset correction: $\widehat{m_n} = m_n/(1-\beta_1^n)$, : $\widehat{v_n} = v_n/(1-\beta_2^n)$

Then the weights are updated according to the formula: $w_{n+1} = w_n - \eta\widehat{m_n}/\sqrt{\widehat{v_n}+\epsilon}$

Since this method is related to maximizing of the value of the absolute gradients, we will consider the error of training and testing under the condition of 100 iterations. Figure 11 shows Fourier spectra and branching diagram, for the function of the learning error from the number of iterations, subject to the application of the optimization method AdamMax, for a homogeneous array of dimensions 3x5, at β_2=0.999 and 1000 iterations. Harmonics can be traced on the Fourier spectra, which prove that when the learning speed is greater than the optimal speed, this method is also characterized by overlearning, which can be associated with the appearance of local minima in the learning error function. The resulting branch diagram for the method AdamMax, provided that the array size is 3x5, in comparison with the Adam

method, under the same conditions, proves that the learning process in the interval *alpha*<0.4 it is more homogeneous.



a) *b*)

Figure **11**: Fourier spectra a) and branching diagram for the learning error function from the number of iterations b), subject to the application of the optimization method AdamMax, for a homogeneous array of dimensions 3x5, at β_2=0.999 and 1000 iterations.

Consider the influence of the optimization parameter β_2 on the learning error at the optimal value of the learning speed for a homogeneous array with a given size of 3x5 digits:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 2.6065e-05÷2.6291e-05;

β_2=0.999, optimal *alpha* = 0.4501; learning error = 2.6026e-05÷2.6285e-05;

β_2=0.9999, optimal *alpha* = 0.4501; training error = 2.6029e-05÷2.6292e-05.

Changing the optimization parameter β_2, within the accuracy of the experiment, does not affect the value of the optimal learning speed. As for the learning error, as it was noted in the paper [12], with an increase in the value β_2 a decrease in its value is observed at first (at β_2=0.999), and then an increase (at β_2=0.9999). As noted in the paper [11], the smallest learning error is observed when the value of the optimization parameter β_2=0.999. So, unlike the method Adam in the AdamMax method, the dependence of the learning error on the value of the parameter β_2 is observed.

4.2. A non-homogeneous array, displaying numbers in a 3x5 array

When studying the influence of the optimization parameter β_2 the following results were obtained for the learning error under the condition of the optimal value of the learning speed for a non-homogeneous array of a given size of 3x5 digits:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 2.0633e-05÷2.0784e-05;

β_2=0.999, optimal *alpha* = 0.4501; learning error = 2.0614e-05÷2.0785e-05;

β_2=0.9999, optimal *alpha* = 0.4501; training error = 2.0635e-05÷2.0786e-05.

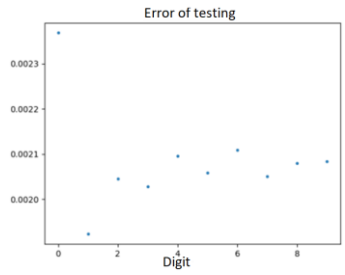
Changing the optimization parameter β_2 within the accuracy of the experiment, both for a homogeneous input array and for a non-homogeneous one, does not affect the value of the optimal learning speed. As for the learning error, here, as noted in the paper [12], with an increase in β_2 first its decrease and then its increase can be traced. According to work [12], the best value of the value of the optimization parameter β_2=0.999. So, in the AdamMax method, the dependence of the learning error on the value of the parameter β_2 can be traced both with a homogeneous and with a non-homogeneous input array. To understand the mechanism of the dependence of the learning error on the parameter β_2, consider the testing error and the dependence of the quality of learning and testing on the number of iterations.

Let's consider how the homogeneity and non-homogeneity of the input array affects the testing error for the 3x5 digit display array. Fig. 12 shows the testing error for different numbers when the optimization parameter is changed β_2 and constant value β_1=0.9, for homogeneous and non-homogeneous input array. Changing the value of the optimization parameter β_2 affects the testing error, both for a homogeneous array and for a non-

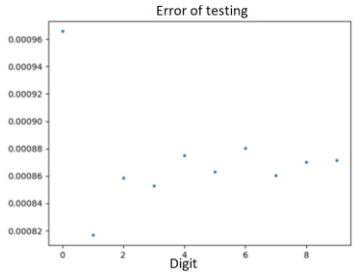
homogeneous array. Increasing the value of β_2 causes a decrease in the testing error by almost an order of magnitude at 100 iterations. For the testing process, it does not depend on the value of β_2 there is a pattern that the testing error for the digits "0" shows a worse result than for other digits.

Comparing the testing error for a homogeneous and non-homogeneous array, according to Fig. 12, a pattern can be observed that the testing error for a non-homogeneous array is almost a third smaller. Comparing the testing error with the learning error, the latter is approximately an order of magnitude smaller. This reflects the dependence of the quality of training (1) and quality of testing (2) on the number of iterations shown in Fig. 13. The interval of a sharp change in their dependence depends both on the optimization parameter β_2 and on the homogeneity of the input array. For a non-homogeneous array, as mentioned above, the training error is smaller than the testing error. Therefore, the learning quality curve is higher than the corresponding test curve. By increasing the parameter β_2 there is a narrowing of the interval in terms of the number of iterations in which a sharp change in the steepness of this dependence takes place. That is, the larger the value of the β_2 parameter, the smaller the number of iterations required to achieve a certain testing accuracy. But compared to a homogeneous array of numbers, with a non-homogeneous array, the quality of training and testing is characterized by a steeper change in the number of iterations. Such a feature of the behavior of both learning and testing errors from the number of iterations proves an important role of large gradients, which improve the stability and learning speed of the neural network when applying the optimization method AdamMax compared to the Adam optimization method.

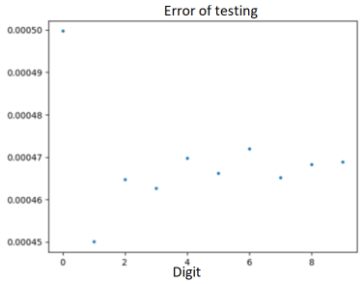
Comparing the test error for different numbers for an optimization method Adam and AdamMax, it can be noted that for the numbers from "2" to "9" the testing error is approximately the same. The largest testing error is observed for the number "0".



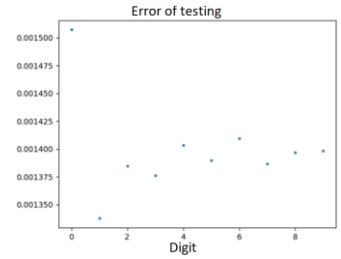
3x5 homogeneous array, β_2=0.99.



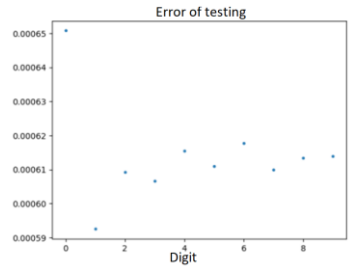
3x5 homogeneous array, β_2=0.999.



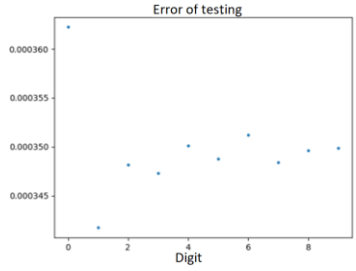
3x5 homogeneous array, β_2=0.9999.



3x5 non-homogeneous array, β_2=0.99.

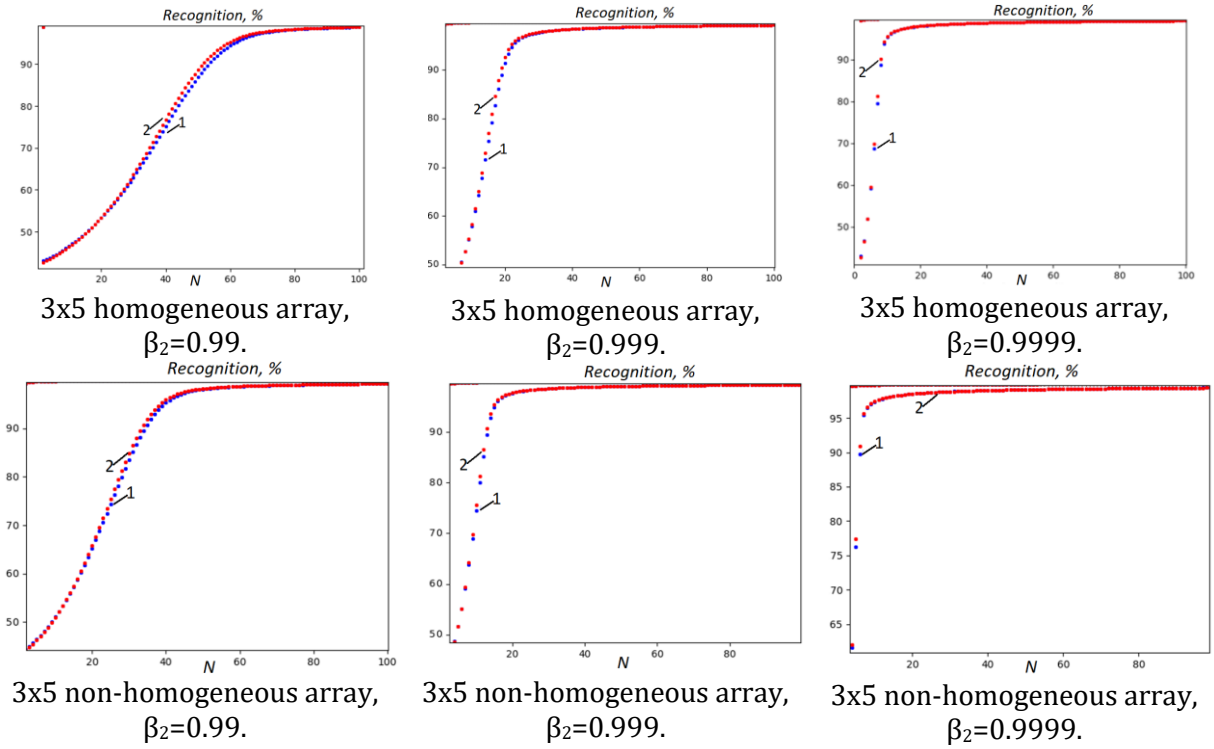


3x5 non-homogeneous array, β_2=0.999.



3x5 non-homogeneous array, β_2=0.9999.

Figure **12**: Testing error for different numbers when changing the optimization parameter β_2 and a constant value of β_1=0.9, for a homogeneous and non-homogeneous input array, provided the method is usedAdamMax, *N*=100 iterations.



3x5 homogeneous array, β_2=0.99. | 3x5 homogeneous array, β_2=0.999. | 3x5 homogeneous array, β_2=0.9999.

3x5 non-homogeneous array, β_2=0.99. | 3x5 non-homogeneous array, β_2=0.999. | 3x5 non-homogeneous array, β_2=0.9999.

Figure **13**: The quality of training (1) and testing (2) from the number of iterations and the optimization parameter β_2 for a homogeneous and non-homogeneous array, provided the method is usedAdamMax, *N*=100 iterations.

4.3. Homogeneous array, representation of numbers in a 4x7 array

Under the influence of the optimization parameter β_2 the following values were obtained for the learning error at the optimal value of the learning speed for a homogeneous array of the target digit of size 4x7:

β_2=0.99, optimal *alpha* = 0.451; learning error = 1.9111e-05÷1.9266e-05;

β_2=0.999, optimal *alpha* = 0.451; learning error = 1.9112e-05÷1.9261e-05;

β_2=0.9999, optimal *alpha* = 0.451; training error = 1.9109e-05÷1.9268e-05.

Changing the optimization parameter β_2 within the accuracy of the experiment for a homogeneous input array does not affect the value of the optimal learning speed. In comparison with the 3x5 digit presentation array, for the 4x7 array there is a slight increase in the optimal learning speed from 0.4501 to 0.451. As for the learning error, here, as noted in [11], within the accuracy of the experiment, with an increase in β_2 first its decrease and then its increase can be traced. The best value of the value of the optimization parameter β_2=0.999, at which the minimum learning error is observed for the given number of iterations. So, for the AdamMax method, the dependence of the learning error on the value of the parameter β_2 can be traced for both the 3x5 and 4x7 number representation arrays, as described in [12].

4.4. Non-homogeneous array, representation of numbers in a 4x7 array

Consider addiction learning errors at the optimal value of the learning speed for a non-homogeneous array of size 4x7 when changing optimization parameter β_2:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 1.5144e-05÷1.5282e-05;

β_2=0.999, optimal *alpha* = 0.4501; training error = 1.5144e-05÷1.5284e-05;

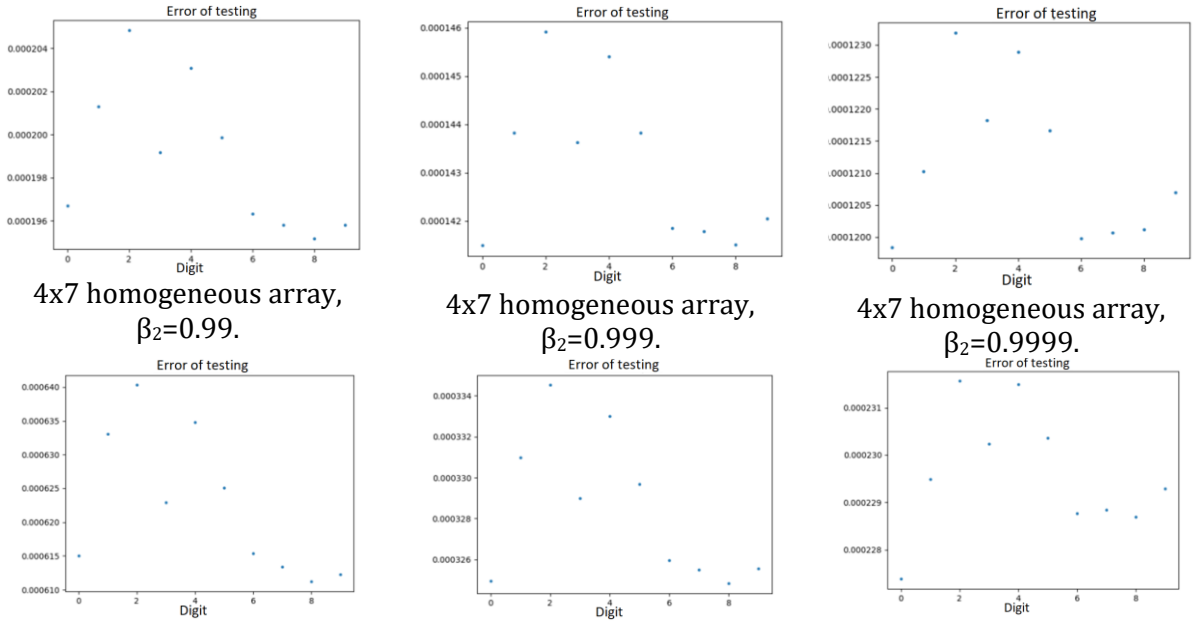
β_2=0.9999, optimal *alpha* = 0.4501; learning error = 1.5154e-05÷1.5283e-05.

Changing the optimization parameter β_2 within the accuracy of the experiment, for a non-homogeneous input array, does not affect the value of the optimal learning speed. As for the

learning error, here, within the accuracy of the experiment, there is a slight increase in the learning error with an increase in the value of β_2. So, in the AdamMax method, the dependence of the learning error on the value of the parameter β_2 can be traced for both a homogeneous and non-homogeneous arrays.

Fig. 14 shows the testing error for different numbers when the optimization parameter β_2 is changed and a constant value of β_1=0.9, for a homogeneous and non-homogeneous input array, provided that the number is represented by a 4x7 array. Changing the value of the optimization parameter β_2 affects the testing error, both for a homogeneous array and for a non-homogeneous array. Increasing the value of β_2 causes a decrease in the testing error by almost two times for a homogeneous array, and three times for a non-homogeneous array, at 100 iterations. For the testing process, it does not depend on the value of β_2 there is a pattern that the testing error for numbers "1", "2", "3", "4", "5" shows a worse result than for other numbers.

Comparing the testing error for a homogeneous and non-homogeneous array, according to Fig. 14, we observe the pattern that the testing error for a non-homogeneous array is almost a third larger and depends on the parameter β_2. If we compare the testing error with the training error, the latter is about an order of magnitude smaller. The dependencies of training quality (1) and testing quality (2) shown in Fig. 15 reflect this. The interval of a sharp change in their dependence depends both on the optimization parameter β_2 and on the homogeneity of the input array. For a non-homogeneous array, as mentioned above, the training error is smaller than the testing error.



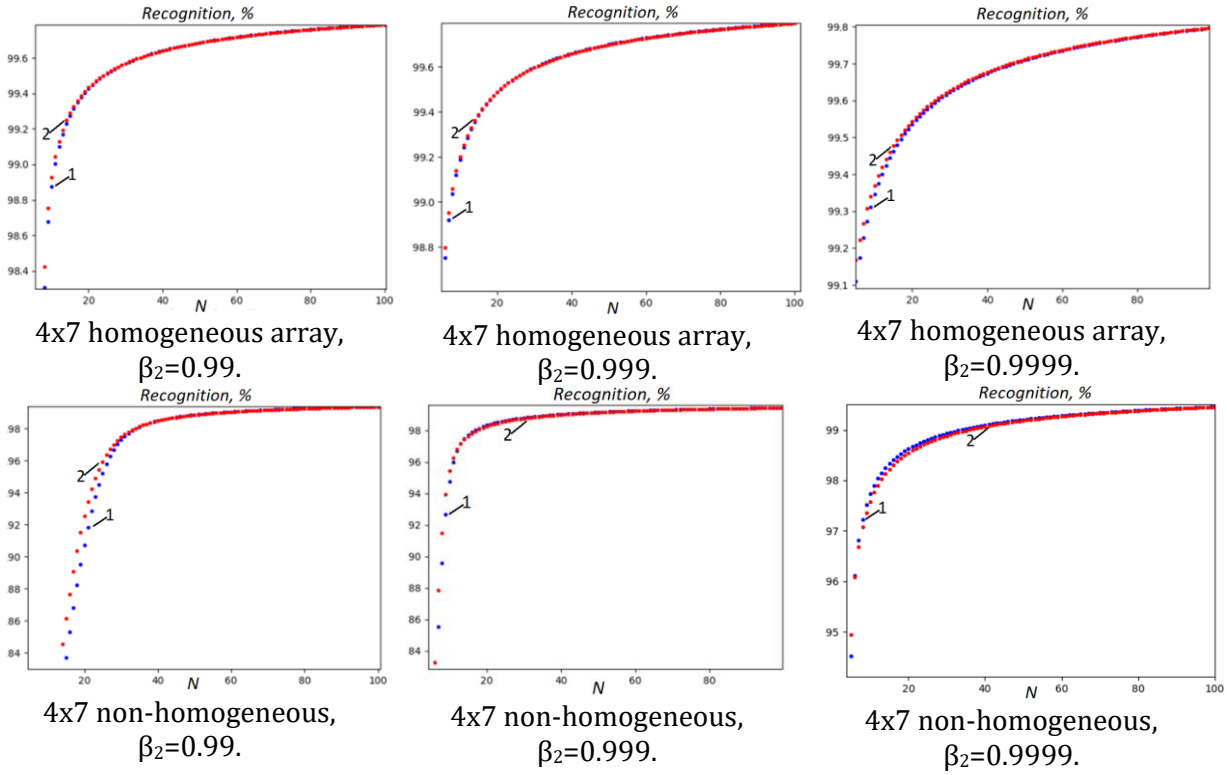
4x7 homogeneous array, β_2=0.99. 4x7 homogeneous array, β_2=0.999. 4x7 homogeneous array, β_2=0.9999.

4x7 non-homogeneous, β_2=0.99. 4x7 non-homogeneous, β_2=0.999. 4x7 non-homogeneous, β_2=0.9999.

Figure **14**: Testing error for different numbers when changing the optimization parameter β_2 and a constant value of β_1=0.9, for a homogeneous and non-homogeneous input array, provided the method is usedAdamMax, *N*=100.

Therefore, the learning quality curve is higher than the corresponding test curve. Although at small iteration values (in the interval of sharp changes in the quality of testing), the opposite trend is observed. By increasing the parameter β_2 for a homogeneous array, the expansion of the iteration interval in which there is a sharp change in the steepness of this dependence can be traced. That is, the larger the value of the β_2 parameter, the greater the number of iterations required to achieve a certain testing accuracy. For a non-homogeneous array, the quality of training and testing when the parameter β_2 increases improves, the interval of sharp changes in each iteration decreases at the beginning, but then increases again. Such a feature of behavior, both learning and testing errors from the number of iterations, as noted above, attests to the

important role of large gradients that improve the stability and learning speed of the neural network when using the optimization method AdamMax compared to the Adam optimization method.



4x7 homogeneous array, β_2=0.99. | 4x7 homogeneous array, β_2=0.999. | 4x7 homogeneous array, β_2=0.9999.

4x7 non-homogeneous, β_2=0.99. | 4x7 non-homogeneous, β_2=0.999. | 4x7 non-homogeneous, β_2=0.9999.

Figure **15**: The quality of training (1) and testing (2) from the number of iterations and the optimization parameter β_2 for a homogeneous and non-homogeneous array, provided the method is usedAdamMax, N=100.

5. The AMSGrad method

5.1. Homogeneous array, representation of numbers in a 3x5 array

AMSGrad (Adaptive Moment Estimation with Squared Gradient) is a variant of the optimization algorithm, which is a modification of Adam (Adaptive Moment Estimation) [12].

The main idea of AMSGrad is to fix the problem of increasing the values of v_n second-order moment estimates in Adam. In regular Adam, v_n increases with each iteration, which can lead to an increase in the learning rate and, as a result, to large changes in the model parameters, which is not always desirable.

AMSGrad solves the problem of excessive v_n growth in Adam by saving the maximum value of v_n from all past steps. That is, AMSGrad allows you to ensure stability of training and more accurate adaptation of the speed of training for each parameter.

The AMSGrad optimization algorithm initializes parameters such as learning rate, exponential averages of the first and second order moment, β_1, β_2, initial values of the first and second order moment m=0, v=0, v_{max}=0 [13].

This method is given by the following formulas:

The calculation of the exponentially weighted mean gradient: $m_n=\beta_1 m_{n-1}+(1-\beta_1)g_n$

The calculation of the exponentially weighted mean square of the gradient: $v_n=\beta_2 v_{n-1}+(1-\beta_2)g_n^2$

$\underline{v}_n=\max(\beta_2 \cdot v_{n-1},|g|)$,

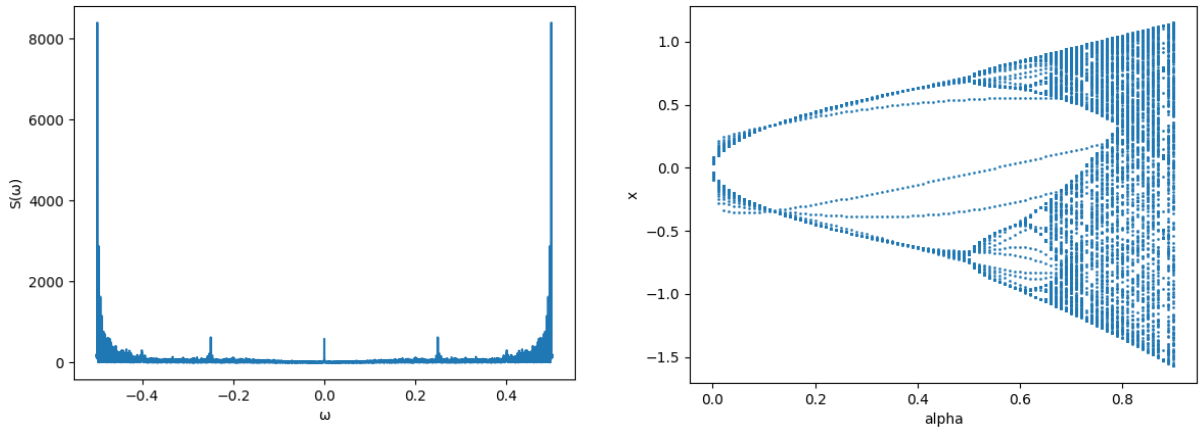
Offset correction: $\widehat{m_n} = m_n/(1-\beta_1^n)$, : $\widehat{v_n} = v_n/(1-\beta_2^n)$

Then the weights are updated according to the formula : $w_{n+1} = w_n - \eta\widehat{m_n}/\sqrt{\widehat{v_n} + \epsilon}$

Model parameter update: $\widehat{v_{max,n}}$ -maximum with $v_{max,n}$ and $\widehat{v_n}$,

ϵ is a small number for numerical stability.

The obtained Fourier spectra (Fig. 16) are similar to the spectra obtained for the Adam and AdamMax optimization methods. These Fourier spectra prove the existence of harmonics, which indicate the presence of a neural network retraining process at a learning speed greater than the optimal speed. The resulting branch diagram for the method AMSGrad provided that the array size is 3x5, in comparison with the Adam method and AdamMax, under the same conditions certifies that the learning process in the interval *alpha*<0.4 it is more homogeneous. Based on the Fourier spectra and branching diagrams, the optimization method AMSGrad also has an inherent chaotic learning mode, which is caused by the appearance of local minima of the learning error function, which in turn are associated with the process of retraining neurons when approaching the global minimum.



a) *b*)

Figure **16**: Fourier spectra a), branching diagram training and testing error depending on the number of iterations b), subject to the application of the AMSGrad optimization method, for a homogeneous array of dimensions 3x5, with β_2=0.999 and 100 iterations.

As for optimization methods Adam and AdamMax, and for the AMSGrad optimization method, we will consider the influence of the optimization parameter β_2 on the learning error at the optimal value of the learning speed for a homogeneous array with a given size of 3x5 digits:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 1.908e-05÷1.9233e-05;

β_2=0.999, optimal *alpha* = 0.4501; training error = 1.9068e-05÷1.9242e-05;

β_2=0.9999, optimal *alpha* = 0.4501; training error = 1.909e-05÷1.9244e-05.

Changing the optimization parameter β_2 within the accuracy of the experiment for a homogeneous input array does not affect the value of the optimal learning speed. As for the learning error, as in the AdamMax method, within the accuracy of the experiment, with an increase in β_2 its increase can be traced. The best value of the value of the optimization parameter in which the minimum learning error is observed for the given number of iterations is β_2=0.999. So, in neural networks using the optimization method AMSGrad to display numbers in a 3x5 array as in the method AdamMax the dependence of the learning error on the value of the parameter β_2 is traced.

5.2. A non-homogeneous array, displaying numbers in a 3x5 array

When considering the influence of the optimization parameter β_2 the following parameters are obtained for the learning error at the optimal value of the learning speed for a non-homogeneous 3x5 array:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 1.5006e-05÷1.519e-05;

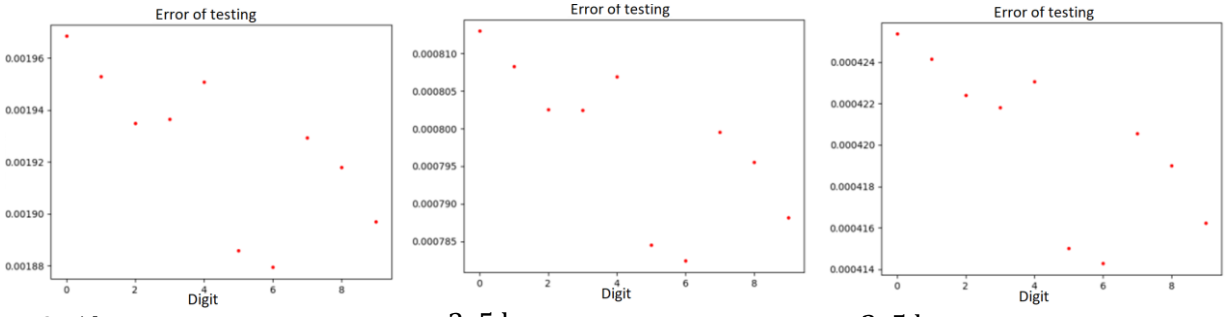
β_2=0.999, optimal *alpha* = 0.4501; learning error = 1.496e-05÷1.5192e-05;

β_2=0.9999, optimal *alpha* = 0.4501; learning error = 1.5075e-05÷1.5196e-05.

Changing the optimization parameter β_2 for a non-homogeneous input array does not affect the value of the optimal learning speed. As for the learning error, here, as noted in [12], within the accuracy of the experiment, with an increase in β_2 first its decrease and then its increase can be traced. The best value of the value of the optimization parameter in which the minimum learning error is observed for a given number of iterations, as well as for a homogeneous array, is β_2=0.999. So, in neural networks using the optimization method AMSGrad the dependence of the learning error on the value of the parameter β_2 is traced.

Consider how the homogeneity and non-homogeneity of the input array affects the testing error for the 3x5 array when using the optimization method AMSGrad. Fig. 17 shows the testing error for different numbers when the optimization parameter β_2 is changed and constant value β_1=0.9, for homogeneous and heterogeneous input array. Changing the value of the optimization parameter β_2 affects the testing error, both for a homogeneous array and for a non-homogeneous array. Increasing the value of β_2 causes a decrease in the testing error by almost an order of magnitude for a homogeneous array, and almost by an order of magnitude for a non-homogeneous array at 100 iterations. For the testing process, regardless of the value of β_2 there is a pattern that the testing error for the digits "0" shows a worse result than for other digits.

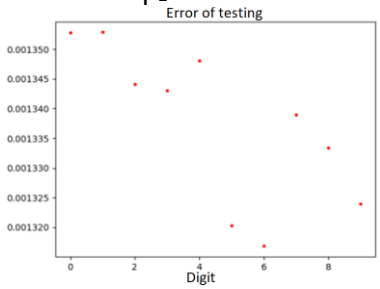
Comparing the testing error for a homogeneous and non-homogeneous array, according to Fig. 17, a pattern can be observed that the testing error for a non-homogeneous array is smaller.



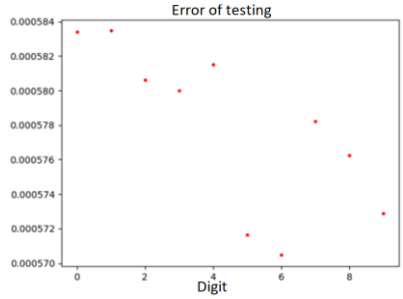
3x5 homogeneous array, β_2=0.99.

3x5 homogeneous array, β_2=0.999.

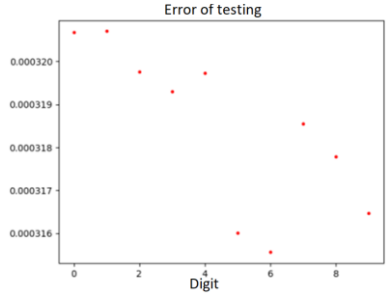
3x5 homogeneous array, β_2=0.9999.



3x5 non-homogeneous array, β_2=0.99.



3x5 non-homogeneous array, β_2=0.999.

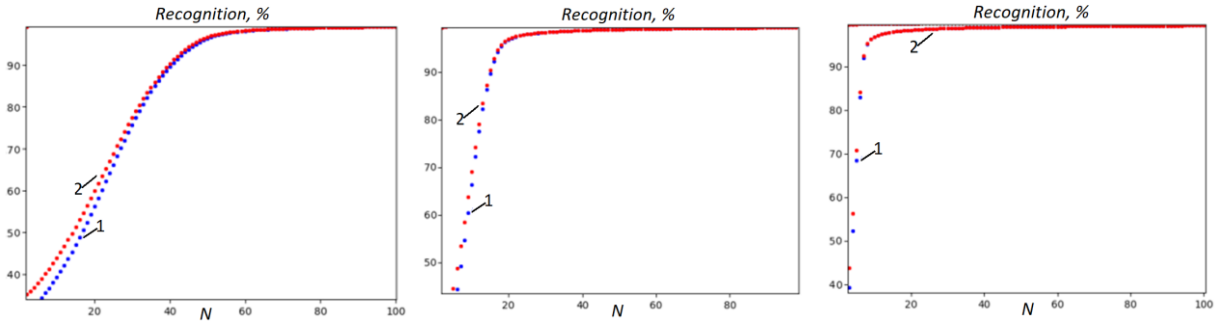


3x5 non-homogeneous array, β_2=0.9999.

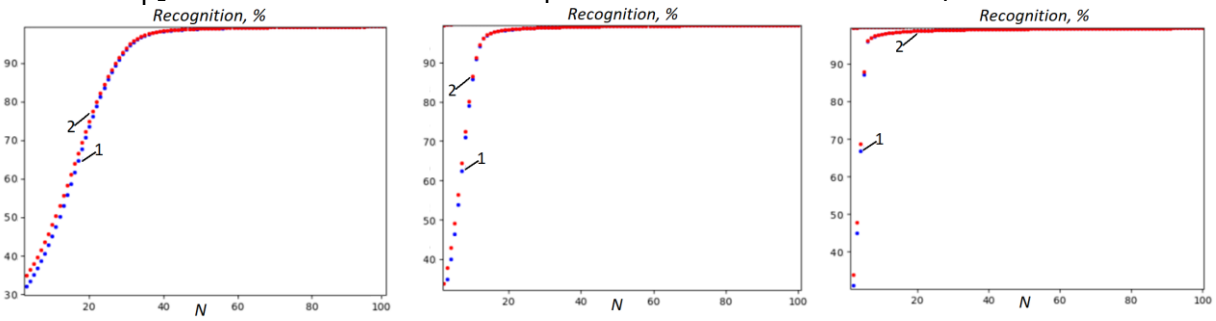
Figure **17**: Testing error for different numbers when changing the optimization parameter β_2 and a constant value of β_1=0.9, for a homogeneous and non-homogeneous input array, provided the method is used AMSGrad, *N*=100.

Comparing the testing error with the learning error, the latter is about an order of magnitude smaller. The dependencies of training quality (1) and testing quality (2) shown in Fig. 18 reflect this. The interval of a sharp change in their dependence on the number of iterations depends both on the optimization parameter β_2 and on the homogeneity of the input array. For a non-homogeneous array, as mentioned above, the training error is smaller than the testing error. Therefore, the curve of the quality of learning is higher than the corresponding curve for testing in the interval outside of its sharp changes. In the interval of iterations, where a sharp change in the quality of learning is observed, this process is the opposite. By increasing the parameter β_2 a

narrowing of the iteration interval in which there is a sharp change in the steepness of this dependence can be traced. That is, the larger the value of the β_2 parameter, the smaller the number of iterations required to achieve a certain testing accuracy. But compared to a homogeneous array of numbers, with a non-homogeneous array, the quality of training and testing is characterized by a steeper change in the number of iterations. Such a feature of the behavior of both learning and testing errors from the number of iterations testifies to the important role of large gradients, which improve the stability and learning speed of the neural network when using the optimization method AMSGrad compared to the Adam optimization method.



3x5 homogeneous array, β_2=0.99. | 3x5 homogeneous array, β_2=0.999. | 3x5 homogeneous array, β_2=0.9999.



3x5 non-homogeneous array, β_2=0.99. | 3x5 non-homogeneous array, β_2=0.999. | 3x5 non-homogeneous array, β_2=0.9999.

Figure **18**: The quality of training (1) and testing (2) from the number of iterations and the optimization parameter β_2 for a homogeneous and non-homogeneous array, provided the method is usedAMSGrad, *N*=100.

5.3. Homogeneous array, representation of numbers in a 4x7 array

Changing the optimization parameter β_2 at the optimal value of the learning speed for a homogeneous array, the given size of digits 4x7 causes the following change in the learning error:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 1.9154e-05÷1.9308e-05;

β_2=0.999, optimal *alpha*= 0.4501; training error = 1.915e-05÷.1.9292e-05;

β_2=0.9999, optimal *alpha*= 0.4501; training error = 1.915e-05÷1.9292e-05.

As for optimization methods Adam, AdamMax and so on for AMSGrad changing the optimization parameter β_2 within the accuracy of the experiment for a non-homogeneous input array does not affect the value of the optimal learning speed. The learning error decreases whit increases. The best value related to the minimum learning error of the value of the optimization parameter β_2 is observed for the given number of iterations is β_2=0.999. So, in the method AMSGrad for both the 3x5 and 4x7 digit display arrays, the dependence of the learning error on the value of the β_2 parameter can be traced.

5.4. Non-homogeneous array, representation of numbers in a 4x7 array

When changing the optimization parameter β_2 under the condition of the optimal value of the learning speed for a non-homogeneous array of a given size of 4x7 digits, the following values of the learning error at 100 iterations were obtained:

β_2=0.99, optimal *alpha* = 0.4501; learning error = 1.5144e-05÷1.526e-05;

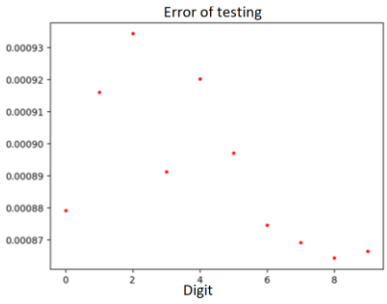
β_2=0.999, optimal *alpha*= 0.4501; training error = 1.5144e-05÷1.5284e-05;

β_2=0.9999, optimal *alpha*= 0.4501; learning error = 1.5154e-05÷1.5283e-05.

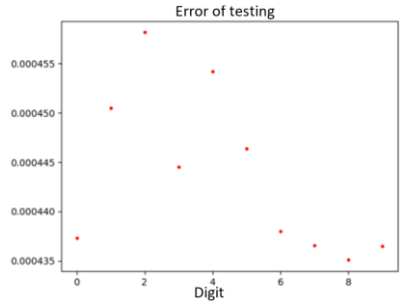
Changing the optimization parameter β_2 within the accuracy of the experiment for a non-homogeneous input array does not affect the value of the optimal learning speed. As for the learning error, with an increase in value β_2 its increase can be traced.

Fig. 19 shows the testing error for different numbers when the optimization parameter β_2 is changed and a constant value of β_1=0.9, for homogeneous and non-homogeneous input array at display numbers in a 4x7 array. Changing the value of the optimization parameter β_2 affects the testing error, both for a homogeneous array and for a non-homogeneous array. Increasing the value of β_2 causes a similar dependence of the testing error for a non-homogeneous array as for a homogeneous one. For the testing process, regardless of the value of β_2 there is a pattern that the testing error for numbers "1", "2", "3", "4" and "5" shows a worse result than for other numbers.

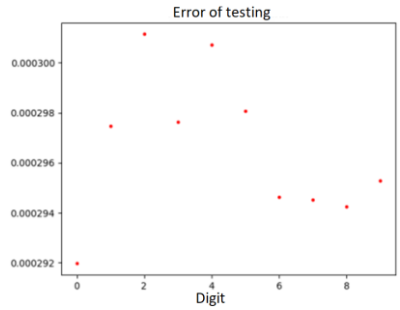
Comparing the testing error for a homogeneous and non-homogeneous array at submission of an array of numbers 4x7 in size, according to Fig. 19, there is a pattern that the testing error is smaller for a non-homogeneous array.



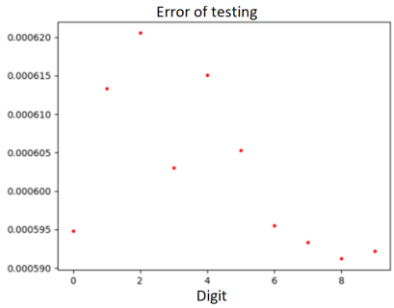
4x7 homogeneous array, β_2=0.99.



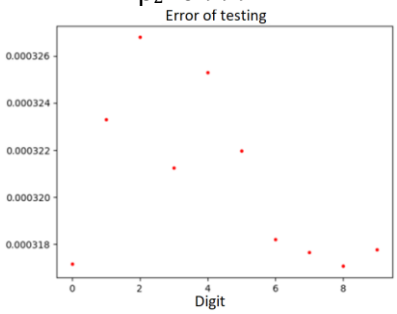
4x7 homogeneous array, β_2=0.999.



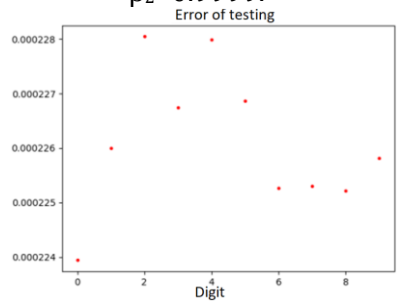
4x7 homogeneous array, β_2=0.9999.



4x7 non-homogeneous array, β_2=0.99.



4x7 non-homogeneous array, β_2=0.999.

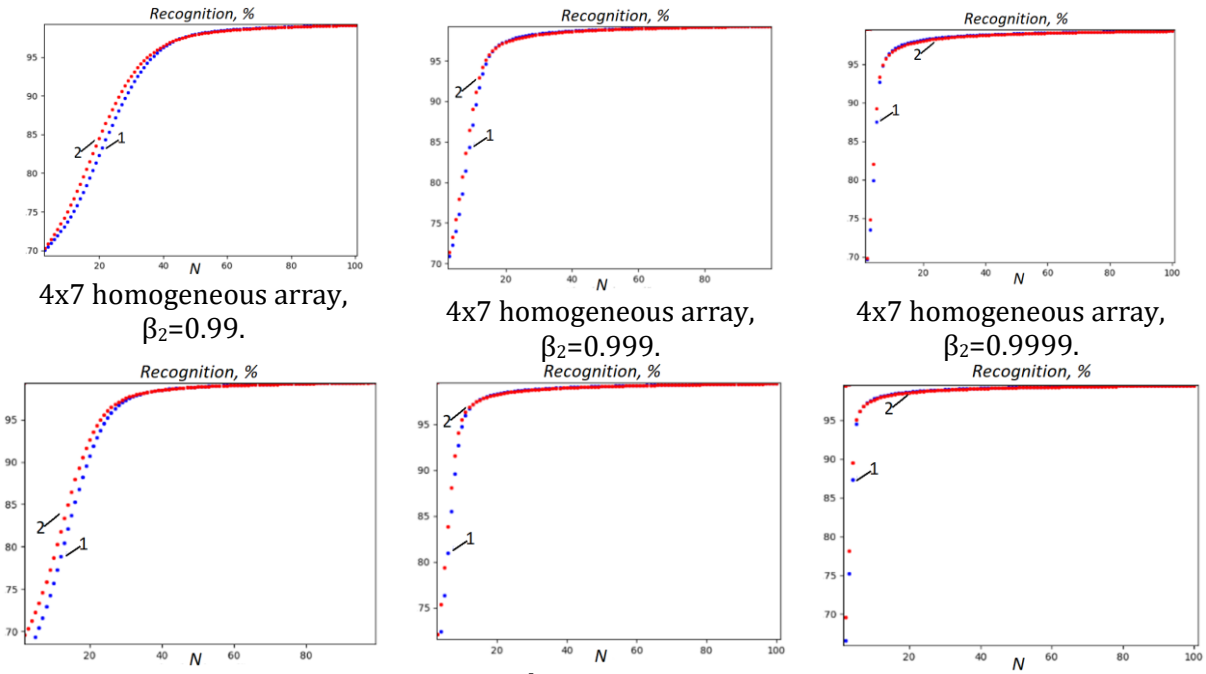


4x7 non-homogeneous array, β_2=0.9999.

Figure **19**: Testing error for different numbers when changing the optimization parameter β_2 and a constant value of β_1=0.9, for a homogeneous and non-homogeneous input array, provided the method is used AMSGrad, *N*=100.

Comparing the testing error with the learning error, the latter is approximately an order of magnitude smaller. The dependencies of training quality (1) and testing quality (2) shown in Fig. 20 reflect this. The interval of a sharp change in their dependence on the number of iterations depends on both the optimization parameter β_2 and the homogeneity of the input array. For a non-homogeneous array, as mentioned above, the training error is smaller than the

testing error. Therefore, the curve of the quality of learning is higher than the corresponding curve for testing in the interval outside of its sharp changes. In the interval of iterations, where a sharp change in the quality of learning is observed, this process is the opposite. By increasing the parameter β_2 a narrowing of the iteration interval in which there is a sharp change in the steepness of this dependence can be traced. That is, the larger the value of the β_2 parameter, the smaller the number of iterations required to achieve a certain testing accuracy. But compared to a homogeneous array of numbers, with a non-homogeneous array, the quality of training and testing is characterized by a steeper change in the number of iterations. Such a feature of the behavior of both learning and testing errors from the number of iterations proves an important role of large gradients, which improve the stability and learning speed of the neural network when using the optimization method AMSGrad compared to the Adam optimization method.



4x7 homogeneous array, β_2=0.99. | 4x7 homogeneous array, β_2=0.999. | 4x7 homogeneous array, β_2=0.9999.

4x7 non-homogeneous array, β_2=0.99. | 4x7 non-homogeneous array, β_2=0.999. | 4x7 non-homogeneous array, β_2=0.9999.

Figure **20**: The quality of training (1) and testing (2) from the number of iterations and the optimization parameter β_2 for a homogeneous and non-homogeneous array, provided the method is used AMSGrad, N=100.

6. Conclusions

Therefore, the conducted studies of learning error and testing error prove that the learning process, depending on the learning speed (learning step), demonstrates the existence of a number of learning modes. Under the condition that the learning rate is less than optimal, a non-learning mode is observed, which can be characterized as a learning mode with an error > 10%. Provided that training takes place in the vicinity of the learning speed values equal to the optimum, a satisfactory learning process with minimal learning error can be observed. The process of relearning takes place when the learning speed is greater than its optimal value. It is accompanied by the appearance of local minima, and therefore, an increase in the learning error. A further increase in the learning speed leads to an increase in the number of neurons that are inherent in the relearning process, and therefore to an increase in the number of local minima. A sharp increase in the number of local minima (which in the first approximation can be described by the process of doubling their number) leads to the emergence of a chaotic state. In this paper, the optimal learning speed was determined from the Fourier spectra of the

learning error function and corresponded to the value of the learning speed at which the Fourier spectra are characterized by the appearance of the first harmonic. The conducted studies of the learning error of neural networks when using the optimization methods of learning AMSGrad, Adam, AdamMax prove that these methods do not affect the value of the optimal learning speed, and it does not depend on the change of the optimization parameter β_2. For all considered optimization methods, it is 0.450. It is approximately equal to the default value used in all known machine learning libraries that use neural networks. In our experimental studies on the influence of the sample and the size of the array of numbers on the value of the optimal learning speed, we demonstrate that it does not change. Although conducting similar studies for an array of handwritten digits, which were set with a display size of 28x28 pixels, showed an increase in its value to 0.5 when using the Adam optimization method. As for the learning error, its value depends on the sample, from the size of the number display array, from the homogeneity of the input array, from the application of optimization learning methods, and the value of the optimization parameters. The change in the slope of the dependence of the learning error on the number of iterations for the considered optimization methods is especially noteworthy. Moving from the Adam optimization method to AdamMax and AMSGrad can be traced to an increase in the steepness of the slope of the dependence of the learning error on the number of iterations. This is due to what is in the methods AdamMax and AMSGrad instead of using the average value of the squares of the gradients, as is done in the Adam method, the maximum value of the absolute values of the gradients for each parameter is used. This allows focusing on large gradients, which improves stability and learning speed. Along with this, it should be noted that the retraining process is associated with the appearance of local minima, which in the final case, when the learning speed changes, lead to a chaotic learning mode of the neural network. The mechanism of transition to the chaotic learning mode of the neural network is related to the process of doubling the number of local minima of the error function. The obtained Fourier spectra of the learning error function, branching diagrams, and the special behavior of the error function when using optimization learning methods prove that the main reason for the appearance of local minima of the error function is the relearning process. As for the smaller value of the learning error for a non-homogeneous input array compared to a homogeneous array, according to the authors, this confirms the important role of large gradients in improved stability and speed of learning.

References

[1] J. Schneider, Cross Validation, 1997. URL: https://www.cs.cmu.edu/~schneide/tut5/node42.html

[2] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens & Z. Wojna, Rethinking the conception architecture for computer vision, Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818-2826. URL: https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.pdf

[3] S. O. Subbotin, Neural networks: theory and practice: teaching. (Ed. O. O. Evenok), Zhytomyr, 2020, 184p.

[4] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting, Journal of Machine Learning Research. Volume 15, 2014, pp. 1929-1958, URL: https://jmlr.org/papers/v15/srivastava14a.html

[5] D. Berrar, Cross-validation. Encyclopedia of Bioinformatics and Computational Biology, Elsevier, Volume 1, 2018, pp. 542–545. URL: https://doi.org/10.1016/B978-0-12-809633-8.20349-X

[6] Y. Yuan, L. Rosasco, A. Caponnetto, On Early Stopping in Gradient Descent Learning. Constructive Approximation. Springer, Volume 26, 2007, pp 289–315. URL: https://doi.org/10.1007/s00365-006-0663-2. ISSN 0176-4276. S2CID 8323954

[7] G. Federico, M. Jones, T. Poggio, Regularization Theory and Neural Networks Architectures. Neural Computation. MIT Press Volume 7 Issue 2, 1995, pp. 219–269. URL: https://doi.org/10.1162/neco.1995.7.2.219 ISSN 0899-7667. S2CID 49743910.
[8] K. Kawaguchi, Effect of Depth and Width on Local Minima in Deep Learning Neural Computation. MIT Press Volume 31 Issue 7, 2019, pp. 1462-1498. URL: https://doi.org/10.1162/neco_a_01195
[9] S. Sveleba, I. Katerynchuk, I. Kuno, N. Sveleba, O. Semotyjuk Investigation of the Transition Mechanism to Chaos in Multilayer Neural Networks 2021 IEEE 4th International Conference on Advanced Information and Communication Technologies (AICT), 2021, pp. 118-121. URL: https://ieeexplore.ieee.org/document/9628919 doi: 10.1109/AICT52120.2021.9628919.
[10] D. P. Kingma, J. B. Adam, A Method for Stochastic Optimization 3rd International Conference for Learning Representations, San Diego, 2015, URL: https://doi.org/10.48550/arXiv.1412.6980
[11] X. Zeng, Z. Zhang and D. Wang, AdaMax Online Training for Speech Recognition, CSLT TECHNICAL REPORT-20150032, 2016, URL: http://www.cslt.org/mediawiki/images/d/df/Adamax_Online_Training_for_Speech_Recognition.pdf
[12] T. T. Phuong, L. T. Phong, On the Convergence Proof of AMSGrad and a New Version IEEE Access, Volume 7, 2019, pp. 61706-61716 URL: https://doi.org/10.48550/arXiv.1904.03590, doi: 10.1109/ACCESS.2019.2916341
[13] J.-K. Wang, X. Li, B. Karimi, P. Li, An Optimistic Acceleration of AMSGrad for Nonconvex Optimization. Proceedings of Machine Learning Research Volume 157, 2021, pp.422-437 URL: https://proceedings.mlr.press/v157/wang21c/wang21c.pdf