UDC 517.2+519.977.58+519.8

V.I. NORKIN
V.M. Glushkov Institute of Cybernetics of the National Academy of Sciences of Ukraine, Kyiv, Ukraine; National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute," Kyiv, Ukraine,
e-mail: *vladimir.norkin@gmail.com*.

STOCHASTIC GENERALIZED GRADIENT METHODS FOR TRAINING NONCONVEX NONSMOOTH NEURAL NETWORKS[1]

Abstract. The paper observes a similarity between the stochastic optimal control of discrete dynamical systems and the learning multilayer neural networks. It focuses on contemporary deep networks with nonconvex nonsmooth loss and activation functions. The machine learning problems are treated as nonconvex nonsmooth stochastic optimization problems. As a model of nonsmooth nonconvex dependences, the so-called generalized-differentiable functions are used. The backpropagation method for calculating stochastic generalized gradients of the learning quality functional for such systems is substantiated basing on Hamilton–Pontryagin formalism. Stochastic generalized gradient learning algorithms are extended for training nonconvex nonsmooth neural networks. The performance of a stochastic generalized gradient algorithm is illustrated by the linear multiclass classification problem.

Keywords: machine learning, deep learning, multilayer neural networks, nonsmooth nonconvex optimization, stochastic optimization, stochastic generalized gradient.

INTRODUCTION

The machine learning problem consists of the identification of parameters of a neural network model, e.g., neural weights, using a set of input-output observations. The training task is formulated as the task of minimizing some smooth loss functional (empirical risk), which measures the average forecast error of the neural network model.

Methods of training (identification) of large neural network models are discussed in many articles and monographs [1–11]. To train deep (i.e., multilayer) neural networks, the stochastic gradient method, and its modifications are mainly used [9–11], adopted from the theory of stochastic approximation [12] and stochastic programming [13–15], since only they are practically applicable for training such networks. The stochastic gradient of the risk functional is a random vector whose mathematical expectation approximates the gradient of the target functional, and the stochastic gradient descent method is an iterative method for changing the desired model parameters in the direction of the stochastic (anti-) gradient.

To solve smooth neural network training problems, the backpropagation of error technique, BackProp method, is widely used [1–8], i.e. a special method for calculating gradients of the target functional over various parameters. The history of discovery, development, and application of the BackProp method was studied in [8]. Nonsmooth machine learning tasks arise when using nonsmooth (module type) indicators of the quality of training, when applying nonsmooth regularizations, and also when using nonsmooth (for example, piecewise linear, ReLU, etc.) activation functions in multilayer neural networks [5, Sec. 6.3.1; 6, Sec. 3.3; 8]. Such functions give rise to essentially nonconvex nonsmooth functionals of the quality of learning, and the question arises of the convergence of the stochastic generalized gradient descent method in solving such problems. This problem has been relatively recently recognized and is already being considered in the literature [16–21]. However, it is usually

[1] The work was partially supported by grant CPEA-LT-2016/10003 funded by the Norwegian Agency for International Cooperation and Quality Enhancement in Higher Education (Diku) and grant 2020.02/0121 of the National Research Foundation of Ukraine.

assumed using the Clarke stochastic subgradients [22] of the optimized functional but the problem of their calculation for deep networks is not profoundly discussed.

In this paper, we extend the BackProp method to calculating stochastic gradients of nonconvex nonsmooth problems for training multilayer neural networks and formulate the method in terms of stochastic generalized gradients of the nonsmooth Hamilton–Pontryagin function. As a model of nonsmooth nonconvex dependencies, we use the so-called generalized-differentiable functions [23, 24]. We also consider an important version of the BackProp method for training the so-called recurrent neural networks, i.e. networks with feedbacks and memory [5, Sec. 10].

In the paper, we show that the convergence of the stochastic generalized gradient method follows from the earlier results of the theory of nonconvex nonsmooth stochastic optimization [24–28]. In [16], as descent directions, Clarke stochastic generalized gradients of the optimized risk functional are used. However, the question remains, what kind of objects the backpropagation method calculates in the case of a nonsmooth nonconvex functional and whether these objects can be used for optimization purposes. It may be supposed that the BackProp method calculates the (stochastic) Clarke subgradients of the optimized function but this takes place only in the case of the so-called subdifferentially regular functions [22, Sec. 2.3], that can be not the case. In this connection, it was proposed in [24, 27–30] to randomize the method of generalized gradient descent, namely, to calculate gradients not at the current iteration point but at a random close point, where the Lipschitz function is almost always differentiable.

Thus, although the problems of learning deep smooth neural networks have been studied for a long time, there are several new aspects related to the nonsmoothness of networks that still require discussion:

nonconvexity and nonsmoothness of the optimized risk functional;

methods for calculating stochastic (generalized) gradients for nonsmooth nonconvex networks;

a convergence of the stochastic gradient method in the nonconvex nonsmooth case;

the method parameters control and the method modifications for solving nonconvex nonsmooth problems;

multiextremal nature of learning tasks;

the possibility of retraining a neural network model.

The purpose of this article is to apply results of the theory of nonconvex nonsmooth stochastic programming to machine learning problems and to discuss the peculiarities of the application of the stochastic (generalized) gradient method for these problems. In particular, we illustrate the application of the stochastic generalized gradients method to the problem of multiclass linear classification.

1. NONCONVEX NONSMOOTH LEARNING PROBLEMS AND CALCULATION OF STOCHASTIC GENERALIZED GRADIENTS

Let us consider a standard neural network model. Let the network consists of m layers of neurons, each layer $i \in \{1, \dots, m\}$ has n_i neurons with numbers $j = 1, \dots, n_i$ and each of them has n_{i-1} inputs and one output. In the initial layer, there are n_1 neurons, each neuron of this layer has n_0 common inputs and one output. The outputs of neurons of each layer go to the inputs of neurons of the next layer. The output layer of the network may consist of one or more neurons.

In the theory of neural networks, the standard mathematical model of neuron (i, j) is some smooth activation function $g_i^j(x_i, w_{ij}, v_{ij})$ (e.g., the logistic sigmoid, the hyperbolic tangent, and etc. [5, Sec. 6.3.2; 6]), which expresses the dependence of the

output signal $x_{(i+1)j}$ of neuron (i, j) on the input signal x_i, for example,

$$x_{(i+1)j} = g_i^j(x_i, w_{ij}, v_{ij}) = (1 + \exp\{-\langle x_i, w_{ij}\rangle - v_{ij}\})^{-1},$$

where $x_i \in \mathrm{R}^{n_{i-1}}$ is a common input of all neurons in layer i; $w_{ij} \in \mathrm{R}^{n_{i-1}}$ and $v_{ij} \in (-\infty, +\infty)$ are the individual weight vector and the activation level of neuron $j \in \{1, \ldots, n_i\}$ in layer i; the expression $\langle x_i, w_{ij}\rangle$ denotes the scalar product of vectors x_i and w_{ij}. The weights w_{ij} and thresholds v_{ij} may satisfy constraints $w_{ij} \in W_{ij}$, $v_{ij} \in V_{ij}$. Here notation like R^n is used for n-dimensional Euclidian vector space.

Nonsmooth machine learning tasks arise when using nonsmooth indicators of the quality of learning, when applying nonsmooth regularization functions, and when using nonsmooth (for example, piecewise linear) activation functions in multilayer neural networks, for example, $g_i^j(x_i, w_{ij}, v_{ij}) = \max\{-1, \min\{1, \langle x_i, w_{ij}\rangle + v_{ij}\}\}$ [5, Sec. 6.3.3].

For example, piecewise linear activation functions are essentially used in the dynamic brain model with positive BSB feedbacks (brain-state-in-box) [31, Sec. 14.10, p. 884].

In [5], the problem of non-differentiability is informally discussed, caused by a nonsmooth activation function, e.g. the function of linear rectification (the positive part of the argument) $g(z) = \max\{0, z\}$ and its generalizations $g(z) = \max\{\alpha z, \beta z\}$, $g(z) = \max_{i \in I} z_i$, and others [5, Sec. 6.3, p. 169; Subsec. 6.3.1, p. 170; Sec. 6.6, p. 197]. The use of piecewise linear activation functions instead of the sigmoidal functions significantly improved the quality of direct neural networks [3, 32].

Note that activation functions themselves can be random, for example, neurons can accidentally fall into the so-called sleep (drop out [5, 6]) state, i.e. produce a zero output signal: $g_i^j(x_i, w_{ij}, v_{ij}, \omega_{ij}) = \omega_{ij} \cdot g_i^j(x_i, w_{ij}, v_{ij})$, where ω_{ij} is an additional random parameter taking values 1 or 0 with probabilities p_{ij} and $1 - p_{ij}$. We assume that the random parameters $\{\omega_{ij}\}$ are independent and combined into a vector $\omega = \{\omega_{ij}\}$ that takes values from a finite set Ω.

In what follows, we assume that the activation functions $g_i^j(x_i, w_{ij}, v_{ij}, \omega_{ij})$ of neurons $j = 1, \ldots, n_i$ in each layer i for any fixed value of ω_{ij} are generalized-differentiable over their variables (x_i, w_{ij}, v_{ij}) in the sense of the following definition, which covers all practical examples.

Definition 1 [23, 24, 33]. A function $f : \mathrm{R}^n \to \mathrm{R}^1$ is called generalized-differentiable at point $\bar{z} \in \mathrm{R}^n$, if in some ε-neighborhood $\{z \in \mathrm{R}^n : \| z - \bar{z} \| < \varepsilon\}$ of the point $\bar{z}$ it is defined an upper semicontinuous at $\bar{z}$ multivalued mapping $\partial f(\cdot)$ with convex compact values $\partial f(z)$ and such that the following expansion holds true:

$$f(z) = f(\bar{z}) + \langle d, z - \bar{z}\rangle + o(\bar{z}, z, d), \tag{1}$$

where $d \in \partial f(z)$, $\langle \cdot, \cdot \rangle$ denotes the scalar product of two vectors, and the remainder term $o(\bar{z}, z, d)$ satisfies the condition: $\lim\limits_{k \to \infty} o(\bar{z}, z^k, d^k) / \| z^k - \bar{z} \| = 0$ for all sequences $d^k \in \partial f(z^k)$, $z^k \to \bar{z}$, as $k \to \infty$. A function f is called generalized-differentiable if it is generalized-differentiable at each point $\bar{z} \in \mathrm{R}^n$; the mapping $\partial f(\cdot)$ is called the generalized gradient mapping of the function f; the set $\partial f(z)$ is called a generalized gradient set of the function $f(\cdot)$ at point z; vectors $d \in \partial f(z)$ are called generalized gradients of the function $f(\cdot)$ at point z.

Properties of generalized-differentiable functions were studied in details in [23, 24, 33]. Any generalized-differentiable function $f(z)$ is locally Lipschitzian and its Clark subdifferential $\partial_C f(z)$ [22] is the minimal (with respect to inclusion) generalized gradient mapping for $f(z)$, i.e. for all $z \in \mathrm{R}^n$ it takes place $\partial f(z) \supseteq \supseteq \partial_C f(z)$ and for almost all $z \in \mathrm{R}^n$ it holds $\partial f(z) = \partial_C f(z)$ [24, Theorem 1.10]. The class of generalized-differentiable functions contains continuously differentiable, convex, concave, weakly convex and weakly concave [26], semismooth [34], and some other piecewise smooth functions [35], and is closed with respect to the operations of maximum, minimum, superposition and mathematical expectation (see [23, 24, 33, 36]).

Suppose there is a (training) set $\{(x_1^s \in \mathrm{R}^{n_0}, y_{m+1}^s \in \mathrm{R}^{n_m}), s = 1, \dots, S\}$ of observations of a network inputs-outputs. The standard training (identification) task for the network with the training quality criterion $\varphi(x_{m+1}^s, y_{m+1}^s)$ (for example, $\varphi(x_{m+1}^s, y_{m+1}^s) = || x_{m+1}^s - y_{m+1}^s ||^2$) is as follows:

$$J(\{w_{ij}\}, \{v_{ij}\}) = \mathbf{E}_\omega \frac{1}{S} \sum_{s=1}^{S} \varphi(x_{m+1}^s, y_{m+1}^s) \to \min_{\{u_{ij} \in W_{ij}\}, \{v_{ij} \in V_{ij}\}}, \tag{2}$$

where $x_{m+1}^s \in \mathrm{R}^{n_k}$ is the vector of outputs of the last network layer for a training example s; $y_{m+1}^s \in \mathrm{R}^{n_k}$ is a known, generally speaking, multidimensional vector of observations of the network outputs; $|| w_{ij} ||$ denotes a norm of the vector w_{ij}; $\mathbf{E}_\omega$ is the mathematical expectation operator over ω; the sequence of layers' outputs $\{x_i^s = (x_{i1}^s, \dots, x_{in_i}^s)^{\mathrm{T}}, i = 2, \dots, m+1\}$ for a given first layer input $x_1^s \in \mathrm{R}^{n_0}$ is given by the relations

$$x_{(i+1)j}^s = g_i^j(x_i^s, w_{ij}, v_{ij}, \omega_{ij}), \ j = 1, \dots, n_i; \ i = 1, \dots, m. \tag{3}$$

The empirical criterion $J(\{w_{ij}\}, \{v_{ij}\})$ in (2) can be interpreted as the mathematical expectation of the random quantity $\varphi(x_{m+1}^s, y_{m+1}^s)$ over a discrete random variable $\theta = (s, \omega)$ that takes values in the set $\Theta = \{1, \dots, S\} \times \Omega$.

In machine learning, together with (2), regularized problems are considered [5, Ch. 7; 6, Sec. 4.1]:

$$J(\{w_{ij}\}, \{v_{ij}\}) = \mathrm{E}_\omega \frac{1}{S} \sum_{s=1}^{S} \varphi(x_{m+1}^s, y_{m+1}^s) +$$

$$+ \sum_{i=1}^{m} \lambda_i \sum_{j=1}^{n_i} (|| w_{ij} ||^\alpha + || v_{ij} ||^\alpha) \to \min_{\{u_{ij} \in W_{ij}\}, \{v_{ij} \in V_{ij}\}} \tag{4}$$

with smooth ($\alpha = 2$) and nonsmooth ($\alpha = 1$) regularizing terms $|| w_{ij} ||^\alpha, || v_{ij} ||^\alpha$, and (penalty) parameters $\lambda_i \geq 0$ for layers $i = 1, \dots, m$. Regularization, on the one hand, improves the conditionality of the problem, and on the other hand, suppresses the influence of excess neurons in the network.

Moreover, possibly, the training examples may contain not only the input and output of a network (for example, features and labels of objects) $\{(x_1^s, y_{m+1}^s), s = 1, \dots, S\}$, but also may include additional intermediate features $y_i^s \in \mathrm{R}^{n_i}$, $i \in I \subset \{2, \dots, m\}$, which can be used to improve the learning of the intermediate layers of the network, i.e. training

examples may take the form of sequences $\{(x_1^s, \{y_i^s, i \in I\}, y_{m+1}^s), s = 1, \ldots, S\}$. Then the criterion of the quality of training takes the following form:

$$J(\{w_{ij}\}, \{v_{ij}\}) = \mathbf{E}_\omega \frac{1}{S} \sum_{s=1}^{S} \sum_{i \in I} \varphi(x_i^s, y_i^s) + \sum_{i=1}^{m} \lambda_i \sum_{j=1}^{n_i} (||w_{ij}||^\alpha + ||\nu_{ij}||^\alpha) +$$

$$+ \mathbf{E}_\omega \frac{1}{S} \sum_{s=1}^{S} \varphi(x_{m+1}^s, y_{m+1}^s) \to \min_{\{u_{ij} \in W_{ij}\}, \{v_{ij} \in V_{ij}\}}. \quad (5)$$

That's why, next, we consider the following general network training task:

$$J(u) = \mathbf{E}_\theta \sum_{i=1}^{m} \varphi_i(x_i(\theta), u_i) + \mathbf{E}_\theta \varphi_{m+1}(x_{m+1}(\theta)) \to \min_{u \in U} \quad (6)$$

subject to constraints (satisfied for all values of the random parameter $\theta \in \Theta$):

$$x_{i+1}(\theta) = g_i(x_i(\theta), u_i, \theta) = \{g_i^j(x_i(\theta), u_{ij}, \theta)\}_{j=1}^{n_i}, \ i = 1, \ldots, m; \ x_1(\theta) \in \mathrm{R}^{n_0}. \quad (7)$$

Here $u = (u_1, \ldots, u_m) \in \mathrm{R}^l$ $(l = \sum_{i=1}^{m} n_i)$ is the vector of all adjusted parameters; $x_i = (x_{i1}, \ldots, x_{in_{i-1}})^{\mathrm{T}}$ is the input vector for neurons in layer i; u_{ij} is the vector of the adjusted parameters of neuron (i, j); $u_i = \{u_{ij}\}_{j=1}^{n_i - 1}$ is the vector of the adjusted parameters of all neurons in layer i; g_i^j is the activation function of neuron j in layer i; $g_i = \{g_i^j\}_{j=1}^{n_i}$ is the vector activation function of the neurons in layer i; $x_1(\theta) \in \mathrm{R}^{n_0}$ is a random vector of input signals to the network; θ is a random vector parameter that defines the distribution of input signals and influences on the propagation of signals through the network; $\mathbf{E}_\theta$ denotes the sign of the mathematical expectation over θ.

In problems (2)–(5) $u_{ij} = (w_{ij}, v_{ij})$, the role of the random parameter θ is played by the random pair $\theta = (s, \omega)$; here $x_1(\theta) = x_1^s$, $\varphi_{m+1}(x_{m+1}(\theta)) = \varphi(x_{m+1}(s, \omega), y_{m+1}^s)$, and

$$\varphi_i(x_i(\theta), u_i, \theta) = \begin{cases} \varphi(x_i(s, \omega), y_i^s) + \lambda_i \sum_{j=1}^{n_i} (||w_{ij}||^\alpha + ||\nu_{ij}||^\alpha), & i \in I, \\ \lambda_i \sum_{j=1}^{n_i} (||w_{ij}||^\alpha + ||\nu_{ij}||^\alpha), & i \notin I. \end{cases}$$

We make the following assumptions.

Assumptions. Suppose that in problem (6), (7) the functions $\varphi_i(x_i, u_i)$, $g_i^j(x_i, u_{ij}, \theta)$, and $\varphi_{m+1}(x_{m+1})$ are generalized-differentiable over the totality of their arguments, respectively, over (x_i, u_i), (x_i, u_{ij}), and x_{m+1} (under fixed θ). Here, the activation function $g_i^j(x_i, u_{ij}, \theta)$ can be of a general form, i.e. optionally, the function g_i^j may depend not on all elements of the vector x_i and the dimension of the vector of the adjustable parameters u_{ij} may not coincide with the dimension of the vector of inputs x_i. The random parameter $\theta \in \Theta$ is a random variable defined on some probability space.

Note that in the literature (see, for example, [16–21]), for the purpose of training neural networks, it is proposed to use (stochastic) Clarke subgradients of the risk

functional $J(u)$ but these subgradients are relatively simple to calculate only for subdifferentially regular Lipschitz functions [22, §2.3, §2.7], and for general nonconvex nonsmooth functions, their calculation may be a problem.

The next theorem exploits a similarity between optimal control problems for discrete dynamical systems and multilayer neural networks, and formalizes a method for calculating stochastic generalized gradients in the problem of training a nonconvex nonsmooth neural network. It extends the well-known method of "backpropagation of the error" (BackProp) [1–5] to nonconvex nonsmooth learning problems.

First, we introduce the following notation. For arbitrary generalized-differentiable (by the totality of variables) vector functions $g_i(x,u) \in \mathrm{R}^{n_i}$ with arguments $x = (x^1, \dots, x^n)^{\mathrm{T}} \in \mathrm{R}^n$, $u = (u^1, \dots, u^l)^{\mathrm{T}} \in \mathrm{R}^l$, we denote the matrices:

$$g_{ix} = \begin{pmatrix} g^1_{ix^1} & \dots & g^1_{ix^n} \\ \dots & \dots & \dots \\ g^{n_i}_{ix^1} & \dots & g^{n_i}_{ix^n} \end{pmatrix} = \begin{pmatrix} g^1_{ix} \\ \dots \\ g^{n_i}_{ix} \end{pmatrix}, \; g_{iu} = \begin{pmatrix} g^1_{iu^1} & \dots & g^1_{iu^l} \\ \dots & \dots & \dots \\ g^{n_i}_{iu^1} & \dots & g^{n_i}_{iu^l} \end{pmatrix} = \begin{pmatrix} g^1_{iu} \\ \dots \\ g^{n_i}_{iu} \end{pmatrix};$$

and for arbitrary generalized-differentiable (over the totality of arguments) scalar functions $f_i(x,u)$, $x \in \mathrm{R}^n$, $u \in \mathrm{R}^l$, and $\varphi_{m+1}(x)$, $x \in \mathrm{R}^n$, let us introduce vectors:

$$(f_{ix})^{\mathrm{T}} = (f_{ix^1}, \dots, f_{ix^n}), \; (f_{iu})^{\mathrm{T}} = (f_{iu^1}, \dots, f_{iu^l}), \; (\varphi_{kx})^{\mathrm{T}} = (\varphi_{kx^1}, \dots, \varphi_{kx^n}),$$

where $(f_{ix}, f_{iu})^{\mathrm{T}}$, $(g^j_{ix}, g^j_{iu})^{\mathrm{T}}$ are some generalized gradients of the functions $f_i(\cdot,\cdot)$, $g^j_i(\cdot,\cdot,\theta)$; $\varphi_{(m+1)x}(\cdot)$ is some generalized gradient of the function φ_{m+1}; the expression $(\cdot)^{\mathrm{T}}$ denotes the transposition of the matrix $(\cdot)$.

Theorem 1. Under the assumptions made, the objective function $J(u)$ of problem (6), (7) is generalized-differentiable with respect to variables $u = (u_1 \in \mathrm{R}^{n_1}, \dots, u_m \in \mathrm{R}^{n_m})$, and the vector

$$h_u(u,\theta) = (h_{1u_1}(x_1(\theta), \psi_1(\theta), u_1), h_{2u_2}(x_2(\theta), \psi_2(\theta), u_2), \dots$$
$$\dots, h_{mu_m}(x_{mm}(\theta), \psi_m(\theta), u_m))^{\mathrm{T}}, \tag{8}$$

is a stochastic generalized gradient of the function $J(u)$ at a given point u, i.e. $\mathbf{E}_\theta h_u(u,\theta) \in \partial J(u)$, where $h_i(x_i, \psi_i, u_i) = f_i(x_i, u_i) + g_i^{\mathrm{T}}(x_i, u_i) \cdot \psi_i$, $i = 1, \dots, m$, is a discrete (over i) Hamilton–Pontryagin function; the vector

$$h_{iu_i}(x_i(\theta), \psi_i(\theta), u_i) = (h_{iu_i^1}(x_i(\theta), \psi_i(\theta), u_i), \dots, h_{iu_i^{n_i}}(x_i(\theta), \psi_i(\theta), u_i))^{\mathrm{T}} =$$
$$= (f_{iu_i^1}(x_i(\theta), u_i) + g^{\mathrm{T}}_{iu_i^1}(x_i(\theta), u_i) \cdot \psi_i(\theta), \dots, f_{iu_i^{n_i}}(x_i(\theta), u_i) +$$
$$+ g^{\mathrm{T}}_{iu_i^{n_i}}(x_i(\theta), u_i) \cdot \psi_i(\theta))^{\mathrm{T}} \in \mathrm{R}^{n_i} \tag{9}$$

is the u_i-component of a generalized gradient of the function $h_i(\cdot, \psi_i, \cdot)$, $i = 1, \dots, m$; $x(\theta) = (x_1(\theta), \dots, x_{m+1}(\theta))$ is a discrete random trajectory of process (7), corresponding to the given sequence of parameters $(u_1, \dots, u_m) = u$ and the random initial data $x_1(\theta) \in R^{n_0}$. Here, the random sequence of auxiliary (conjugate) vector functions $(\psi_1(\theta), \dots, \psi_m(\theta)) = \psi(\theta)$ is determined through the backpropagation equations (adopted from the Pontryagin maximum principle):

$$\psi_m(\theta) = \varphi_{(m+1)x_{m+1}}(x_{m+1}(\theta)),$$

$$\psi_{i-1}(\theta) = h_{ix_i}(x_i(\theta), \psi_i(\theta), u_i) = f_{ix_i}(x_i(\theta), u_i) + g^{\mathrm{T}}_{ix_i}(x_i(\theta), u_i) \cdot \psi_i(\theta), \quad (10)$$

$$i = m, m-1, \ldots, 2;$$

$(f_{ix}(x_i(\theta), u_i), f_{iu}(x_i(\theta), u_i))^{\mathrm{T}}$, $(g^j_{ix}(x_i(\theta), u_i), g^j_{iu}(x_i(\theta), u_i))^{\mathrm{T}}$ are some generalized gradients of the functions $f_i(\cdot,\cdot)$, $g^j_i(\cdot,\cdot,\theta)$ at the point $(x_i(\theta), u_i)$ and $\varphi_{(m+1)x}(x_{m+1}(\theta))$ is some generalized gradient of the function φ_{m+1} at the point $x_{m+1}(\theta)$, which are used in (9), (10).

Proof. Note that process (7) can be formally treated as a stochastic dynamic system in discrete time $i = 1, \ldots, m+1$ with sates x_i, control parameters u_i, a given initial state $x_1(\theta)$, and the optimality criterion (6). The stochasticity of system (7) is generated by the random input $x_1(\theta)$, a random mechanism of dropping out of neurons, and, possibly, by other factors. So the present theorem is a particular case of a similar Theorem 7 from [37], established for discrete stochastic dynamic system. Similar to the proofs of Theorems 6, 7 from [37] and using relations (7), the vectors x_i, $i = 2, \ldots, m+1$, can be sequentially excluded from the formulation of the optimization problem (6). Then under the sign of summation in (6) there remains some complex composite function

$$f(u, \theta) = \sum_{i=1}^{m} \varphi_i(\widetilde{x}_i(u_1, \ldots, u_{i-1}, \theta), u_i) + \varphi_{m+1}(\widetilde{x}_{m+1}(u_1, \ldots, u_m, \theta)),$$

which depends on optimization variables u and where $\widetilde{x}_i(u_1, \ldots, u_{i-1}, \theta)$ are complex compound functions of their arguments. And since the class of generalized-differentiable functions is closed with respect to compositions, then under the made assumptions this function $f(u, \theta)$ becomes generalized-differentiable with respect to u for each $\theta \in \Theta$. The mathematical expectation $\mathbf{E}_\theta$ (in this case, summation) does not move out from the class of generalized-differentiable functions, therefore the function $J(u) = \mathbf{E}_\theta f(u, \theta)$ is also generalized-differentiable with respect to u. Now, similar to the proofs of Theorems 6–8 from [37], applying the rules of differentiation of the sum, the chain rule of differentiation of complex generalized-differentiable functions (which are analogues to the rules of differentiation of smooth and convex functions) [23, 24, 33], and introducing auxiliary variables (10), after some algebraic transformations (see [37, Theorem 6]), we obtain formula (8) for stochastic generalized gradients $h_u(u, \theta)$ of the function $J(u)$. The proof is complete.

Formulas (8), (9) use procedure (7) of direct calculation of the trajectory of motion $x_1, x_2, \ldots, x_{m+1}$ and the reverse calculation (10) of auxiliary conjugate variables $\psi_m, \ldots, \psi_1$, essentially adopted from the Pontryagin maximum principle [38, 39]. Thus, the vector $h_u(u, \theta)$ is a stochastic generalized gradient of the function $J(u)$ such that $\mathbf{E}_\theta h_u(u, \theta) \in \partial J(u)$, and it can be used in stochastic generalized gradient methods for minimizing the nonsmooth functional $J(u)$. Note that the set of generalized gradients $\partial J(u)$ may be wider than the Clarke subdifferential $\partial_C J(u)$, and it may turn out that $\mathbf{E}_\theta h_u(u, \theta) \notin \partial_C J(u)$. To take this possibility into account, in the next section we modify the standard stochastic gradient descent method by introducing an artificial randomization into it.

Neural networks may have a complex and heterogeneous structure. However, introducing additional dummy neurons, such networks can be reduced to a canonical multilayer form and for them, one can apply the formulas of Theorem 1.

Let us consider an important special case of networks, in which the adjusted parameters are the same for each layer. For example, a network can consist of identical neurons, or some identical neurons are added to each layer in the already-trained network, or the network consists of duplicates of the same layer. Then, similarly to (6), (7), the training task consists in solving the following problem:

$$J(u) = \mathbf{E}_\theta \sum_{i=1}^{m} \varphi_i(x_i(\theta), u) + \mathbf{E}_\theta \varphi_{m+1}(x_{m+1}(\theta)) \to \min_{u=(u_1,\dots,u_l)\in U}, \quad (11)$$

$$x_{i+1}(\theta) = g_i(x_i(\theta), u, \theta) = \{g_i^j(x_i(\theta), u, \theta)\}_{j=1}^{n_i}, \ i = 1, \dots, m; \ x_1(\theta) \in \mathrm{R}^{n_0}. \quad (12)$$

Theorem 2. Under the assumptions made, the objective function $J(u)$ of problem (11), (12) is generalized-differentiable with respect to variables $u = (u_1, \dots, u_l)$, and the vector

$$\left(\sum_{i=1}^{m} h_i(x_i(\theta), \psi_i(\theta), u) \right)_u = \left(\sum_{i=1}^{m} h_{iu_1}(x_i(\theta), \psi_i(\theta), u), \sum_{i=1}^{m} h_{iu_2}(x_i(\theta), \psi_i(\theta), u), \dots \right.$$

$$\left. \dots, \sum_{i=1}^{m} h_{iu_l}(x_i(\theta), \psi_i(\theta), u) \right)^{\mathrm{T}}, \quad (13)$$

is a stochastic generalized gradient of the function $J(u)$ at point u, i.e., $\mathbf{E}_\theta \left(\sum_{i=1}^{m} h_i(x_i(\theta), \psi_i(\theta), u) \right)_u \in \partial J(u)$, where $h_i(x_i, \psi_i, u) = f_i(x_i, u) + g_i^{\mathrm{T}}(x_i, u) \cdot \psi_i$, $i = 1, \dots, m$, is a discrete (over i) Hamilton–Pontryagin function; $x(\theta) = (x_1(\theta), \dots$ $\dots, x_{m+1}(\theta))$ is a discrete random trajectory of process (12) that corresponds to the vector parameter u and the random initial data $x_1(\theta) \in R^{n_0}$. Here, the random sequence of auxiliary (conjugate) vector functions $(\psi_1, \dots, \psi_m) = \psi$ is determined through the backpropagation equations:

$$\psi_{i-1}(\theta) = h_{ix_i}(x_i(\theta), \psi_i(\theta), u) = f_{ix_i}(x_i(\theta), u) + g_{ix_i}^{\mathrm{T}}(x_i(\theta), u) \cdot \psi_i(\theta),$$

$$i = m, m-1, \dots, 2, \ \psi_m(\theta) = \varphi_{(m+1)x_{m+1}}(x_{m+1}(\theta)).$$

This theorem is an analogue of Theorem 6 from [37] and is proved similarly.

2. THE METHOD OF STOCHASTIC GENERALIZED GRADIENT DESCENT AND ITS VARIANTS

The stochastic gradient descent method is the main method for training deep neural networks, firstly, because of enormous dimensions of such networks and, secondly, due to the regularizing properties of the method. The properties of the stochastic gradient method and its modifications were studied in details in cases of smooth and convex optimized functions [9–11, 13–15]. In this article, we analyze this method as applied to nonsmooth nonconvex problems of machine learning.

In the notation and assumptions of the previous section, the learning task (6), (7) consists in optimizing the complex function of the mathematical expectation $J(u) = \mathbf{E}_\theta f(u, \theta)$, where

$$f(u, \theta) = \sum_{i=1}^{m} \varphi_i(\tilde{x}_i(u_1, \dots, u_{i-1}, \theta), u_i) + \varphi_{m+1}(\tilde{x}_{m+1}(u_1, \dots, u_m, \theta)).$$

As indicated in Theorem 1, we can assume that the integrand $f(u,\theta)$ is generalized-differentiable with respect to variables $u=(u_1,\dots,u_m)$ for each fixed θ; then the function $J(u)=\mathbf{E}_\theta f(u,\theta)$ is also generalized-differentiable with respect to u, and its stochastic generalized gradients can be calculated by formulas (8), (10).

We now consider the (randomized) method of stochastic generalized gradient descent to minimize on a convex set U a generalized-differentiable mathematical expectation function,

$$J(u)=\mathbf{E}_\theta f(u,\theta)\to\min_{u\in U}. \tag{14}$$

The method has the form:

$$u^{k+1}\in\Pi_U(u^k-\rho_k d^k)=\arg\min_{u\in U}\left\{\langle d^k,u-u^k\rangle+\frac{1}{2\rho_k}||u-u^k||^2\right\}, \tag{15}$$

$$d^k=d(\tilde{u}^k,\theta^k)\in\partial_u f(\tilde{u}^k,\theta^k),\ ||\tilde{u}^k-u^k||\le\delta_k,\ k=0,1,\dots, \tag{16}$$

where k denotes an iteration number; $\Pi_U(\cdot)$ is the projection operator on a convex feasible set U; $d(u,\theta)$ is a (u,θ)-measurable section (see [36] for details) of a generalized gradient mapping $\partial_u f(u,\theta)$ of the generalized-differentiable random function $f(\cdot,\theta)$; $\{\theta^k,\ k=0,1,\dots\}$ are independent identically distributed observations of the random variable θ; points $\{\tilde{u}^k\}$ are randomly sampled from the sets $\{u:||u-u^k||\le\delta_k\}$; non-negative quantities ρ_k, δ_k can depend on $\{u^0,\dots,u^{k-1}\}$ but must be measurable with respect to the σ-algebra $\sigma\{u^0,\dots,u^{k-1}\}$ and with probability one must satisfy the conditions:

$$\lim_{k\to\infty}\delta_k=\lim_{k\to\infty}\rho_k=0,\ \sum_{k=0}^{\infty}\rho_k=+\infty,\ \sum_{k=0}^{\infty}\rho_k^2<+\infty. \tag{17}$$

Here, the randomization consists in calculating the current generalized gradient $d(\tilde{u}^k,\theta^k)\in\partial_u f(\tilde{u}^k,\theta^k)$ not at the current iteration u^k but at a random close point $\tilde{u}^k$, where $||\tilde{u}^k-u^k||\le\delta_k$. A similar idea of randomization of the generalized gradient descent algorithm is used in [24, 29, 30]. If $\delta_k\equiv 0$, then method (15)–(17) turns into the usual method of stochastic generalized gradient descent. If $U=\mathrm{R}^n$, then the projection operation $\Pi_U(\cdot)$ in method (15) is absent. We denote $N_U(u)$ the cone of normals to the set U at a given point u.

Theorem 3 (convergence with probability 1 of the non-convex randomized method of stochastic generalized gradients [28, Sec. 2; 29, Theorem 5]. Under conditions (17), for almost all trajectories of the process (15), (16), the minimum (in terms of the function J) limit points of the sequence $\{u^k\}$ belong to the set $U_C^*=\{u\in U:0\in\hat{\partial}_C J(u)+N_U(u)\}$ of points satisfying the Clarke's necessary optimality conditions [22], and all limit points of the numerical sequence $\{J(u^k)\}$ constitute an interval in the set $J_C^*=J(U_C^*)$. If the set J_C^* does not contain intervals (for example, J_C^* is finite or countable), then all limit points of the sequence $\{u^k\}$ belong to the connected subset of U_C^*, and there is a limit $\lim_{k\to\infty}J(u^k)\in J_C^*$.

If in algorithm (15)–(17) all $\delta_k\equiv 0$, then the statement of Theorem 3 holds for the set $U^*=\{u\in U:0\in\partial J(u)+N_U(u)\}$ [27, Theorem 5.1]. The convergence of the randomized method of generalized gradient descent for deterministic problems is shown in [27, Remark 4.2; 28; 40, Remark 4.2]. Similar results on the convergence of

the (nonrandomized) method of stochastic generalized gradient descent using generalized Clarke gradients for piecewise smooth Lipschitz (Whitney stratifiable) functions were recently obtained (by another, differential inclusion method) in [16].

Many other stochastic methods of convex optimization were considered in [6, 7, 9, 11] (methods with averaging the trajectory, averaging generalized gradients, ravine step methods, heavy ball, and others). In [24], these methods were extended to problems of non-convex nonsmooth stochastic optimization, in particular, the stochastic ravine step method for solving problem (14) has the form:

$$y^0 = u^0,$$

$$y^{k+1} = \Pi_U(u^k - \rho_k d^k),$$

$$u^{k+1} = y^{k+1} + \lambda_k(y^{k+1} - y^k),\ k = 0, 1, \dots,$$

where $d^k = d(u^k, \theta^k) \in \partial_u f(u^k, \theta^k)$; $\{\theta^k, k = 0, 1, \dots\}$ are independent identically distributed observations of the random variable θ; numbers ρ_k, λ_k satisfy conditions:

$$0 \le \rho_{k+1} \le \rho_k \le \rho,\ \sum_{k=0}^{+\infty} \rho_k = +\infty,\ \sum_{k=0}^{+\infty} \rho_k^2 < +\infty;\ 0 \le \lambda_k \le \lambda < 1.$$

This is a stochastic analog of the deterministic ravine method [41], which, when optimizing smooth convex functions, has a high convergence rate of the order $O(1/k^2)$ [42]. A geometric interpretation of the method shows that it moves (descends) along the "gullies" of the minimized function or the boundary of the feasible region. In [6, 7, 11, 43–45], adaptive step adjustments in stochastic gradient optimization methods are considered.

The randomized method also admits the following interpretation [24]. We introduce the so-called smoothed functions:

$$J_k(u) = \frac{1}{V_{\delta_k}} \int\limits_{\|\tilde{u} - u\| \le \delta_k} J(\tilde{u})\, d\tilde{u}, \tag{18}$$

where V_{δ_k} is the volume of the δ_k-neighborhood of zero. If we introduce a random vector $\tilde{u}^k$, uniformly distributed in the δ_k-neighborhood of point u, then the smoothed function $J_k(u)$ and its gradient $\nabla J_k(u)$ can be represented, respectively, in the form $J_k(u) = \tilde{\mathbf{E}} J(\tilde{u}^k)$ and $\nabla J_k(u) = \tilde{\mathbf{E}} \partial J(\tilde{u}^k)$, where $\tilde{\mathbf{E}}$ denotes the mathematical expectation over $\tilde{u}^k$, $\tilde{\mathbf{E}} \partial J(\tilde{u}^k)$ denotes the mathematical expectation over $\tilde{u}^k$ of the random multivalued mapping $(u, \tilde{u}^k) \to \partial J(\tilde{u}^k)$ under fixed u.

Thus, the randomization in method (15)–(17) plays a threefold role: on the one hand, it allows us to narrow the convergence set of the generalized gradient method to the set $U_C^* \subset U^*$, and on the other hand, it provides to method (15)–(17) some global properties due to the fact that it minimizes the sequence of smoothed functions $J_k(u)$. Besides, the randomization prevents the method to sticking at critical points that are not local minima. In case $\delta_k = \delta$ the randomized generalized gradient method (15)–(17) becomes a stochastic gradient method for minimizing the same non-changing smoothed function $J_k(u)$. To strengthen global properties of method (15)–(17) it is possible to use the estimate of the gradient of the smoothed function (18) using several independent realizations of a random point $\tilde{u}$ from δ_k-vicinity of the current point u^k.

Note that the use of smoothed functions with the norm $||u|| = \max_i |u_i|$ allows us to construct stochastic finite-difference minimization methods, i.e. methods without calculating subgradients [24, 46]. A review of randomized gradient algorithms for optimizing smooth functions is also available in [47].

3. THE METHOD OF STOCHASTIC GENERALIZED GRADIENTS IN THE LINEAR CLASSIFICATION PROBLEM

As an illustration of the application of the method of stochastic generalized gradient descent for training nonsmooth neural networks, we consider the classical problem of multiple classifications of objects based on a family of precedents [4, 7, 48–50]. For classification purposes, the methods of nonsmooth optimization were used in [51–53], and the method of stochastic generalized gradients was used in [7]. The linear classifier can be used as an output layer of a deep neural network in transfer learning models [5, Sec. 15.2; 6]. In this section, we use the stochastic generalized gradient method to solve the multiclass classification problem with a new learning/loss function.

The abstract setting of the problem has the following form. Let a graph be given and fixed, which is used for the representation of classification objects. Such objects will be the subsets of edges (or vertices, or both) of this graph. Each object can have several forms, i.e. it can be represented by different collections of edges. In this model, edges can be interpreted as features of the objects, and each instance of an object can be interpreted as a set of specific features. Obviously, instead of edges, graph vertices can be used to represent objects. Let the edges of the graph be numbered. We will encode objects with (0, 1)-strings with the number of elements equal to the number of edges. Moreover, 0 or 1 at a certain position in the string means that this edge is not used or is used to represent the object. Thus, a set of training examples is encoded by a set of (0, 1)-strings with labels belonging to one or another class. A graphical interpretation is used to visualize objects. Representation of objects in the form of (0, 1)-lines can be noisy, i.e. instead of 0 and 1 in the string at their positions can be random numbers with values from zero to one. By noising, we can expand the original training set arbitrarily largely. Examples may include unclassified objects as a separate class, therefore, without loss of generality, we assume that all examples belong to one or another class. In numerical experiments, noisy codes of stylized numbers and letters were used.

Let us denote $x = (x_0 = 1, x_1, \ldots, x_n)$ the vector of features of a presented object, where for convenience the feature $x_0 = 1$ is introduced, and $x_i \in \{0, 1\}$ or $x_i \in [0, 1]$. We introduce a numerical $(m \times n)$-matrix $W = \{w_{ij}\}$, where m is the number of classes, with rows $w_i = (w_{i0}, w_{i1}, \ldots, w_{in})$, which must be determined based on training examples. The linear classification method $k(x)$ determines the class of the object x under the found weights W according to the formula:

$$k(x) \in \arg\max_{1 \le i \le m} \langle x, w_i \rangle .$$

The matrix of weights W can be found in many ways. For example, it can be found by minimizing the smooth convex cross entropy function:

$$F(W) = -\sum_{s=1}^{S} \ln \frac{\exp \langle w_{i_s}, x^s \rangle}{\sum_{i=1}^{m} \exp \langle w_i, x^s \rangle} \to \min_W , \tag{19}$$

where S is the number of training examples; $x^s = (x_0^s, x_1^s, \ldots, x_n^s)$ denotes the feature vector of the example s; i_s is the known class number (label) for the example s. For comparison, we also use the classic learning function:

$$\Omega(W)=\sum_{s=1}^{S}\left(\max_{1\le i\le m,\, i\ne i_s}\langle w_i,x^s\rangle-\langle w_{i_s},x^s\rangle\right)\to\min_W. \qquad (20)$$

This article also proposes to use the following nonsmooth convex learning function:

$$\Phi(W)=\sum_{s=1}^{S}\left(\max_{1\le i\le m}\langle w_i,x^s\rangle-\langle w_{i_s},x^s\rangle\right)\to\min_W. \qquad (21)$$

Compared to $\Omega(W)$, the function $\Phi(W)$ is advantageous in that its optimal (zero) value is known (for linearly separable data). The advantage of the function $\Phi(W)$ compared to $F(W)$ is in its simplicity. It's obvious that $\Phi(W)\ge 0$. If the classes are linearly separable, i.e. there exists a matrix W^* such that $i_s\in\arg\max_{1\le i\le m}\langle w_i^*,x^s\rangle$ for all $s=1,\dots,S$, then $\min_W\Phi(W)=0$. Note that the set of minima of the function $\Phi(W)$ is not empty and includes non-trivial subsets of the space of all $(m\times n)$-matrices, in particular, all matrices such that their rows coincide belong to the set of minima of the function $\Phi(W)$. Therefore, not all minima of the problem $\min_W\Phi(W)$ are suitable for solving the classification problem. The tasks of minimizing the functions $F(W)$, $\Phi(W)$, and $\Omega(W)$ can be solved by the method of stochastic subgradients. Denote $\Phi^s(W)=\max_{1\le i\le m}\langle w_i,x^s\rangle-\langle w_{i_s},x^s\rangle=\langle w_{i_s^*},x^s\rangle-\langle w_{i_s},x^s\rangle$, where the index i_s^* is such that $\max_{1\le i\le m}\langle w_i,x^s\rangle=\langle w_{i_s^*},x^s\rangle$. Subgradients of the function $\Phi^s(W)$ have the following components:

$$\Phi^s_{w_{ij}}(W)=\begin{cases}x_j^s & \text{if } i=i_s^* \text{ and } i\ne i_s,\\ -x_j^s & \text{if } i=i_s \text{ and } i\ne i_s^*,\\ 0, & \text{if } i=i_s^*=i_s \text{ or } (i\ne i_s^* \text{ and } i\ne i_s);\end{cases}\qquad i=1,\dots,m;\ j=0,1,\dots,n.$$

If an index s (the number of a training example) is chosen randomly and equiprobably, then the vector with the components

$$(\Phi^s_{w_{10}}(W),\dots,\Phi^s_{w_{1n}}(W),\Phi^s_{w_{20}}(W),\dots$$

$$\dots,\Phi^s_{w_{2n}}(W),\dots,\Phi^s_{w_{m0}}(W),\dots,\Phi^s_{w_{mn}}(W))^{\mathrm{T}}=\Phi^s_w(W)$$

is called the stochastic subgradient of the function $\Phi(\cdot)$ at a point W.

The randomized method with averaging stochastic subgradients for solving problem (21) has the form:

$$w_{ij}^{k+1}=w_{ij}^k-\rho_k\cdot\frac{1}{L}\sum_{l=1}^{L}\Phi^{s_l}_{w_{ij}}(\widetilde{W}^{kl}),\ i=1,\dots,m,\ j=0,1,\dots,n, \qquad (22)$$

where $k=0,1,\dots$ denotes the iteration number of the algorithm; $\{s^k,k=0,1,\dots\}$ are independently and equally likely taken numbers of training examples; $W^k=\{w_{ij}^k\}$ is $(m\times n)$-matrix with components w_{ij}^k; $\widetilde{W}^{kl}=\{\widetilde{w}_{ij}^{kl}\}$ is $(m\times n)$-matrix with components $\widetilde{w}_{ij}^{kl}$; $\{\Phi^{s_k}_{w_{ij}}(\widetilde{W}^{kl})\}$ are components of the the stochastic generalized gradient $\Phi^{s_k}_w(\widetilde{W}^{kl})$ at a random point $\widetilde{W}^{kl}$ such that $||W^k-\widetilde{W}^{kl}||\le\delta_k$; W^0 is an initial set of weights; L is the number of subgradients averaged at each iteration. In numerical experiments, it was set $\rho_k=\rho_0/k^{1/2}$, $\delta_k=\rho_0/k^{1/3}$. In the classical stochastic subgradient method, it is assumed that $\delta_k\equiv 0$, $L=1$. When solving problems (19), (20) in method (22), instead of $\Phi^{s_k}_w$ the subgradients of the functions F and Ω are used.

Figures 1–6 show the results of training linear classifiers by the stochastic subgradient method using learning functions $\Phi(W), \Omega(W)$, and $F(W)$. The numbers of train and test examples were 230, initial weights were either zero or random, steps

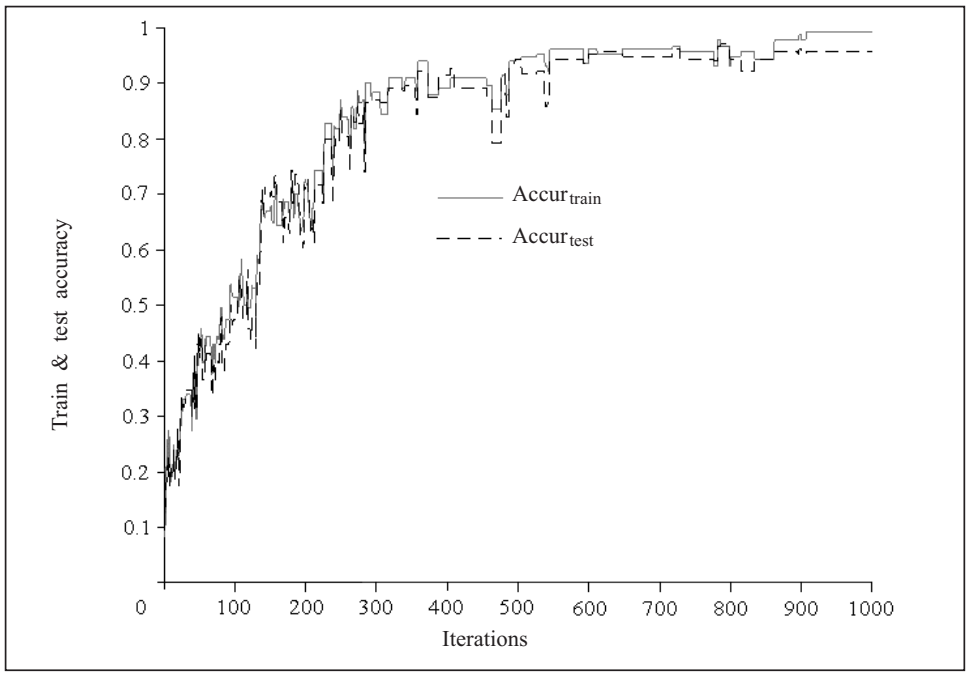

Fig. 1. The performance of the stochastic subgradient method on the function Φ under zero initial weights

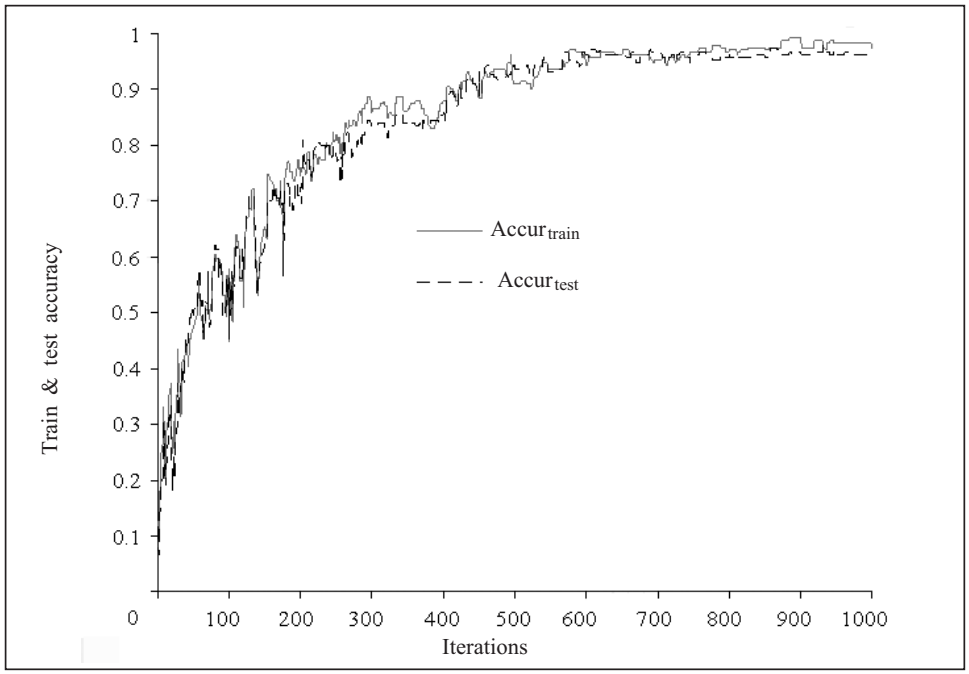

Fig. 2. The performance of the randomized stochastic subgradient method on the function Φ under zero initial weights

$\rho_t = t^{-0.5}$. In Figs. 1, 3, 6 $\delta_t = 0$, $L = 1$. In Figs. 2, 4, 5 $\delta_t = t^{-0.3}$, $L = 10$. These figures show typical examples of performance of the constructed linear classifiers, i.e. they show the fractions of correctly recognized examples in the training and test

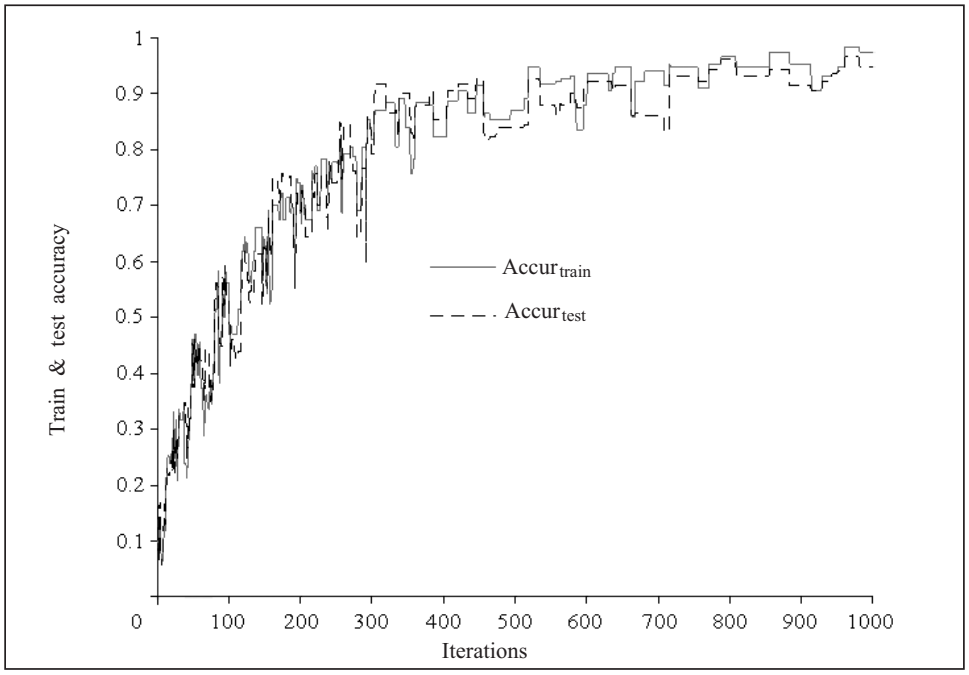

Fig. 3. The performance of the stochastic subgradient method on the function Φ under some random initial weights

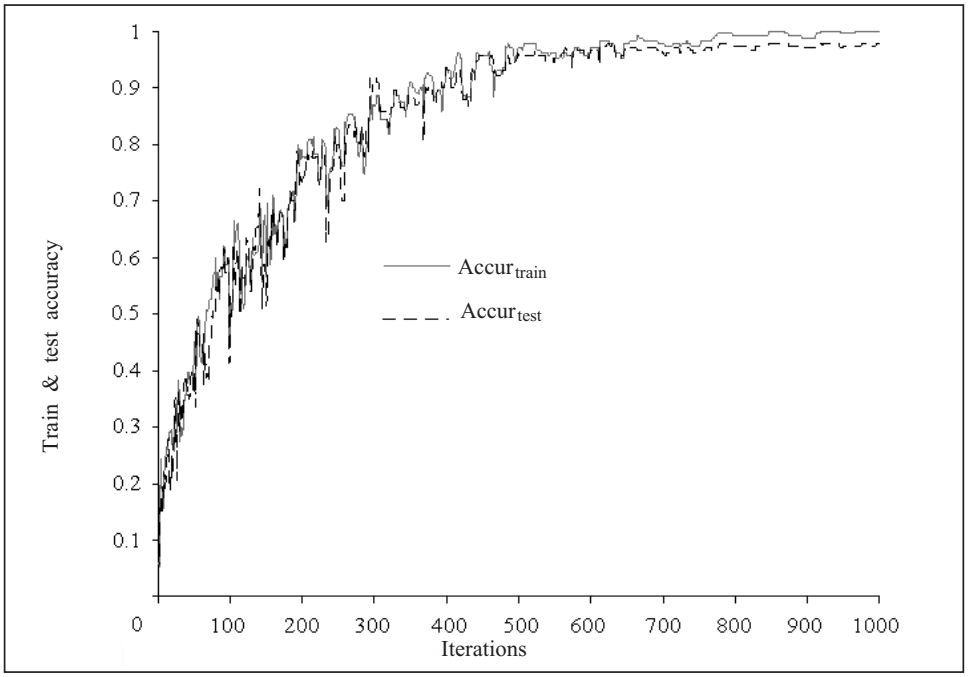

Fig. 4. The performance of the randomized stochastic subgradient method on the function Φ under random initial weights

samples, as a function of the number of iterations of the stochastic subgradient method. The results of the numerical experiments indicate that the linear classification with the function Φ is more effective than using classical functions F and Ω.

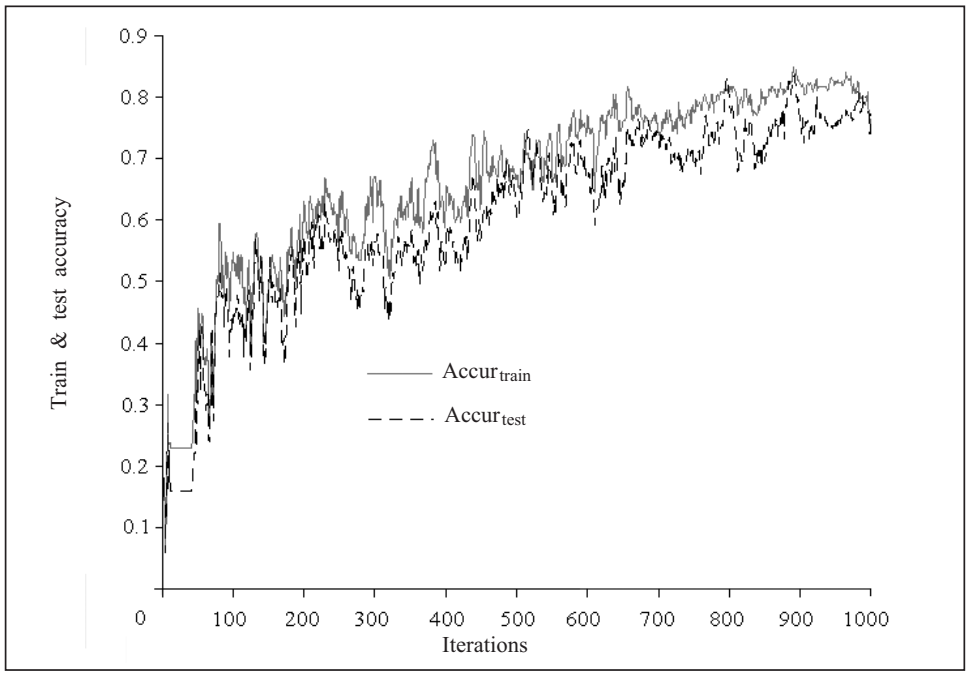

Fig. 5. The performance of the stochastic subgradient method on the function Ω under zero initial weights

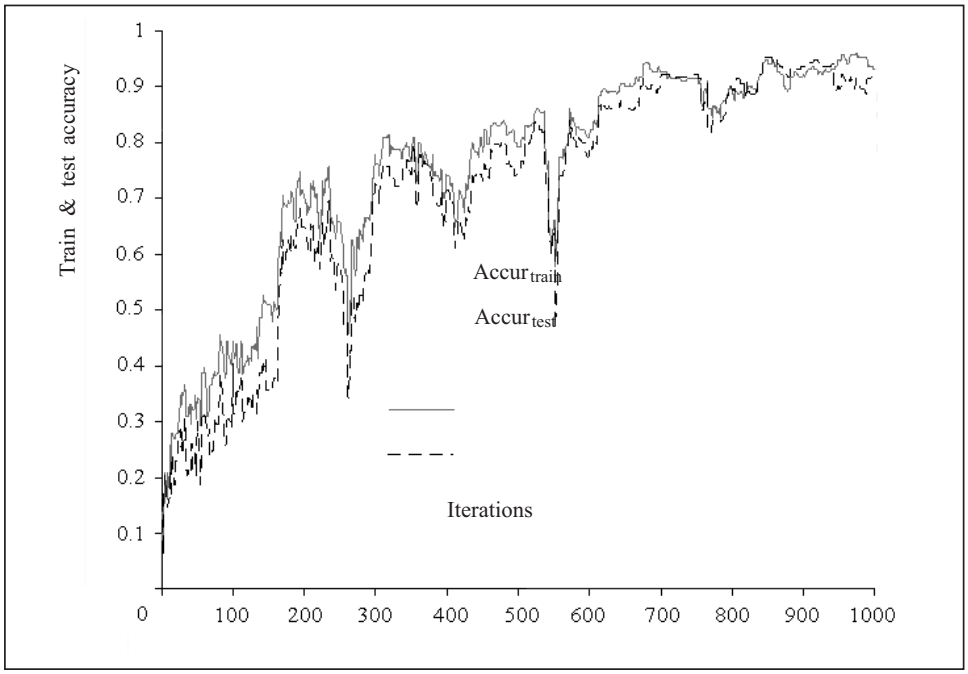

Fig. 6. The performance of the randomized stochastic subgradient method on the function F under zero initial weights

CONCLUSIONS

In the present work, the following results were obtained:

the well-known method of backpropagation of errors is extended to nonconvex nonsmooth machine learning problems;

the randomized stochastic generalized gradient method is substantiated for the training nonsmooth nonconvex deep neural networks;

for the linear classification problem, a variant of the error function is proposed and the advantages of the method of stochastic generalized gradients are demonstrated.

It is of interest to extend the methods of block-coordinate [11] and asynchronous [17, 20] stochastic gradient descent to general nonconvex nonsmooth machine learning problems, which are smoothed out through artificial randomization.

REFERENCES

1. Rumelhart D.E., Hinton G.E., Williams R.J. Learning representations by back-propagating errors. *Nature*. 1986. Vol. 323, P. 533–536. https://doi.org/10.1038/323533a0.
2. LeCun Y.A, Bottou L., Orr G.B., Muller K.-R. Efficient BackProp. In: NN: Tricks of the trade, 2nd ed.. *LNCS*, Vol. 7700. Montavon G., Orr G.B., Muller K.-R. (Eds.). Berlin; Heidelberg: Springer-Verlag, 2012. P. 9–48.
3. LeCun Y., Bengio Y., Hinton G. Deep learning. 436. *Nature*. 2015. Vol. 521. P. 436–444. https://doi.org/10.1038/nature14539.
4. Flach P. Machine learning. The art and science of algorithms that make sense of data. Cambridge University Press, 2012. 409 p.
5. Goodfellow I., Bengio Y., Courville A. Deep learning. MIT Press, 2016. 800 p.
6. Nikolenko S., Kadurin A., Arhangelskaia E. Deep learning. St. Petersburg: Piter, 2018. 480 p. (In Russian).
7. Vorontsov K.V. Machine learning (a year course). URL: http://www.machinelearning.ru/wiki/ (Last accessed: 11.07.2019).
8. Schmidhuber J. Deep learning in neural networks: An overview. *Neural Networks.* 2015. Vol. 61. P. 85–117. http://dx.doi.org/10.1016/j.neunet.2014.09.003.
9. Bottou L., Curtisy F.E., Nocedalz J. Optimization methods for large-scale machine learning. *SIAM Rev.* 2018. Vol. 60, N 2. P. 223–311. https://doi.org/10.1137/16M1080173.
10. Zhang C., Liao Q., Rakhlin A., Miranda B., Golowich N., Poggio T. Theory of deep learning IIb: Optimization properties of SGD. CBMM Memo. N 072. Cambridge, MA: Center for Brains, Minds, and Machines, McGovern Institute for Brain Research, Massachusetts Institute of Technology, 2018. 9 p. arXiv:1801.02254v1 [cs.LG] 7 Jan 2018.
11. Newton D., Yousefian F., Pasupathy R. (2018). Stochastic gradient descent: Recent trends. INFORMS TutORials in operations research. 2018. P. 193–220. https://doi.org/10.1287/educ.2018.0191.
12. Robbins H. Monro S. A stochastic approximation method. *The Annals of Mathematical Statistics.* 1951. Vol. 22, N 3. P. 400–407.
13. Ermoliev Y.M. Methods of stochastic programming. Moscow: Nauka, 1976. 240 p. (In Russian).
14. Nemirovski A., Juditsky A., Lan G., Shapiro A. Robust stochastic approximation approach to stochastic programming. *SIAM J. on Optimization*. 2009. Vol. 19, N 4. P. 1574-1609.
15. Shapiro A., Dentcheva D., Ruszczyński A. Lectures on stochastic programming: Modeling and theory. Philadelphia: SIAM, 2009. 546 p.
16. Davis D., Drusvyatskiy D., Kakade S, Lee J.D. Stochastic subgradient method converges on tame functions. *Found. Comput. Math.* 2020. Vol. 20, Iss. 1. P. 119–154. https://doi.org/10.1007/s10208-018-09409-5.
17. Zhu R., Niu D., Li Z. Asynchronous stochastic proximal methods for nonconvex nonsmooth optimization. arXiv:1802.08880v3 [cs.LG] 15 Sep 2018.

18. Li Z., Li J. A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. In: Advances in Neural Information Processing Systems. Bengio S., Wallach H., Larochelle H., Grauman K., Cesa-Bianchi N., Garnett R. (Eds.). 2018. P. 5564–5574.
19. Majewski S., Miasojedow B., Moulines E. Analysis of nonsmooth stochastic approximation: the diferential inclusion approach. arXiv:1805.01916v1 [math.OC] 4 May 2018.
20. Kungurtsev V., Egan M., Chatterjee B., Alistarh D. Asynchronous stochastic subgradient methods for general nonsmooth nonconvex optimization. arXiv:1905.11845 May 28, 2019.
21. Davis D., Drusvyatskiy D. Stochastic model-based minimization of weakly convex functions. *SIAM J. Optim.* 2019. Vol. 29, Iss.1. P. 207–239. https://doi.org/10.1137/18M1178244.
22. Clarke F.H. Optimization and nonsmooth analysis. Ser. Classics in Applied Mathematics. Vol. 5. Philadelphia: Society for Industrial and Applied Mathematics (SIAM), 1990. 320 p.
23. Norkin V.I. Generalized-differentiable functions. *Cybernetics*. 1980. Vol. 16, N 1. P. 10–12. https://doi.org/10.1007/BF01099354.
24. Mikhalevich V.S., Gupal A.M., Norkin V.I. Methods of nonconvex optimization. Moscow: Nauka, 1987. 280 p. (In Russian).
25. Nurminskii E.A. Minimization of nondifferentiable functions in the presence of noise. *Cybernetics*. 1974. Vol. 10, N 4. P. 619–621. https://doi.org/10.1007/BF01071541.
26 Nurminski E.A. Numerical methods for solving stochastic minimax problems. Kyiv: Naukova dumka, 1979. 176 p. (In Russian).
27. Ermoliev Y.M., Norkin V.I. Stochastic generalized gradient method for solving nonconvex nonsmooth stochastic optimization problems. *Cybernetics and Systems Analysis.* 1998. Vol. 34, N 2. P. 196–215. https://doi.org/10.1007/BF02742069.
28. Norkin V.I. Stochastic methods for solving nonconvex stochastic optimization problems and their applications: Extended abstract ... Doctor thesis. Kyiv, 1998. 27 p. (In Ukrainian). URL: http://library.nuft.edu.ua/ebook/file/01.05.01%20Norkin%20VI.pdf.
29. Ermoliev Y.M., Norkin V.I. Solution of nonconvex nonsmooth stochastic optimization problems. *Cybernetics and Systems Analysis*. 2003. Vol. 39, N 5. P. 701–715. https://doi.org/10.1023/B:CASA.0000012091.84864.65.
30. Burke J., Lewis A., Overton M. A robust gradient sampling algorithm for nonsmooth nonconvex optimization. *SIAM J. Opt.* 2005. Vol. 15, Iss. 3. P. 751–779.
31. Haykin S. Neural networks. A comprehensive foundation. 2nd ed. Prentice Hall, 2006. 842 p.
32. Jarrett K., Kavukcuoglu K., Ranzato M., LeCun Y. What is the best multi-stage architecture for object recognition? *Proc. 2009 IEEE 12th International Conference on Computer Vision (ICCV).* (29 Sept.-2 Oct. 2009, Kyoto, Japan). Kyoto, 2009. P. 2146–2153. https://doi.org/10.1109/iccv.2009.5459469.
33. Norkin V.I. Nonlocal minimization algorithms of nondifferentiable functions. *Cybernetics*. 1978. Vol. 14, N 5. P. 704–707. https://doi.org/10.1007/BF01069307.
34. Mifflin R. An algorithm for constrained optimization with semi-smooth functions. *Math. Oper. Res.* 1977. Vol. 2, N 2. P. 191–207.
35. Bolte J., Daniilidis A., Lewis A. Tame functions are semismooth. *Math. Program*. 2009. Vol. 117, Iss. 1–2. P. 5–19. https://doi.org/10.1007/s10107-007-0166-9.
36. Norkin V.I. Stochastic generalized-differentiable functions in the problem of nonconvex nonsmooth stochastic optimization. *Cybernetics*. 1986. Vol. 22, N 6. P. 804–809. https://doi.org/10.1007/BF01068698.
37. Norkin V.I. Generalized gradients in problems of dynamic optimization, optimal control, and machine learning. *Cybernetics and Systems Analysis.* 2020. Vol. 56, N 2. P. 243–258. https://doi.org/10.1007/s10559-020-00240-x.
38. Pontryagin L.S., Boltianski V.G., Gamkelidze R.V., Mischenko E.F. The mathematical theory of optimal processes. New York; London: Interscience Publishers, 1962. 360 p.
39. Boltianski V.G. Optimal control of discrete systems. Moscow: Nauka, 1973. 446 p. (In Russian).

40. Ermoliev Yu.M., Norkin V.I. Stochastic generalized gradient method with application to insurance risk management. Interim Report IR-97-021. Laxenburg (Austria): Int. Inst. for Appl. Syst. Anal., 1997. 19 p. URL: http://pure.iiasa.ac.at/id/eprint/5270/1/IR-97-021.pdf.
41. Gel'fand I.M., Tzeitlin M.L. The principle of the nonlocal search in automatic optimization systems. *Dokl. Akad. Nauk SSSR*. 1961. Vol. 137(2). P. 295–298.
42. Nesterov Y. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. S*oviet Math. Dokl.* 1983. Vol. 27(2). P. 372–376.
43. Urjas'ev S.P. Step control for direct stochastic-programming methods. *Cybernetics*. 1980. Vol. 16, N 6. P. 886-890. https://doi.org/10.1007/BF01069063.
44. Urjas'ev S.P. Stochastic quasi-gradient algorithms with adaptively controlled parameters. Working paper WP-86-32. Laxenburg (Austria): Int. Inst. for Appl. Syst. Anal., 1986. 27 p. URL: http://pure.iiasa.ac.at/id/eprint/2827/1/WP-86-032.pdf.
45. Urjas'ev S.P. Adaptive algorithms of stochastic optimization and game theory. Moscow: Nauka, 1990. 184 p. (In Russian).
46. Gupal A.M. Stochastic methods for solution of nonsmooth extremal problems. Kyiv: Naukova dumka, 1979. 150 p. (In Russian).
47. Granichin O.N., Polyak B.T. Randomized algorithms of optimization and estimation under almost arbitrary noise. Moscow: Nauka, 2003. 293 p. (In Russian).
48. Vapnik V.N. Statistical learning theory. New York: Wiley & Sons, 1998. 768 p.
49. Shlezinger M., Glavach V. Ten lectures on the statistical and structural recognition. Kyiv: Nakova dumka, 2004. 545 p. (In Russian).
50. Yuan G.-X., Ho C.-H., Lin C.-J. Recent advances of large-scale linear classification. *Proc. of the IEEE*. 2012. Vol. 100, Iss. 9. P. 2584–2603. https://doi.org/10.1109/JPROC.2012.2188013.
51. Laptin Yu.P., Zhuravlev Yu.I., Vinogradov A.P. Empirical risk minimization and problems of constructing linear classifiers. *Cybernetics and Systems Analysis*. 2011. Vol. 47, N 4. P. 640–648. https://doi.org/10.1007/s10559-011-9344-0.
52. Zhuravlev Yu.I., Laptin Yu.P., Vinogradov A.P., Zhurbenko N.G., Lykhovyd O.P., Berezovskyi O.A. Linear classifiers and selection of informative features. *Pattern Recognition and Image Analysis.* 2017. Vol. 27, N 3. P. 426–432. https://doi.org/10.1134/S1054661817030336.
53. Zhuravlev Yu.I., Laptin Yu.P., Vinogradov A.P. A comparison of some approaches to classification problems, and possibilities to construct optimal solutions efficiently. *Pattern Recognition and Image Analysis*. 2014. Vol. 24, N 2. P. 189–195. https://doi.org/10.1134/S1054661814020175.

В.І. Норкін

СТОХАСТИЧНІ УЗАГАЛЬНЕНІ ГРАДІЄНТНІ МЕТОДИ НАВЧАННЯ НЕОПУКЛИХ НЕГЛАДКИХ НЕЙРОННИХ МЕРЕЖ

Анотація. У статті відмічено подібність між стохастичним оптимальним керуванням дискретними динамічними системами та навчанням багатошарових нейронних мереж. Роботу зосереджено на дослідженні сучасних глибоких мереж з неопуклими негладкими функціями втрат та активації. Проблеми машинного навчання розглянуто як неопуклі негладкі задачі стохастичної оптимізації. Як модель негладких неопуклих залежностей використано так звані узагальнено диференційовні функції. Метод зворотного обчислення стохастичних узагальнених градієнтів функціоналу якості навчання для таких систем обґрунтовано на основі формалізму Гамільтона–Понтрягіна. Стохастичні узагальнені алгоритми градієнтного навчання поширено для навчання неопуклих негладких нейронних мереж. Ефективність стохастичного узагальненого градієнтного алгоритму проілюстровано прикладом лінійної багатокласової класифікаційної задачі.

Ключові слова: машинне навчання, глибоке навчання, багатошарові нейронні мережі, негладка неопукла оптимізація, стохастична оптимізація, стохастичний узагальнений градієнт.

Надійшла до редакції 05.04.2021