

Nataliya Shakhovska,  
Sergio Montenegro,  
Yannick Estève,  
Sergey Subbotin,  
Natalia Kryvinska,  
Ivan Izonin



# Informatics & Data- Driven Medicine

Proceedings of the 1st International Workshop  
IDDM 2018

Lviv, Ukraine  
November 28-30,  
2018

Shakhovska N., Montenegro S., Estève Y., Subbotin S., Kryvinska N., Izonin I., (Eds.):  
The 1<sup>st</sup> International Workshop on Informatics & Data-Driven Medicine (*IDDM 2018*).  
Lviv, Ukraine, November 28-30, 2018, CEUR-WS.org, online

This volume represents the proceedings of the 1<sup>st</sup> International Workshop on Informatics & Data-Driven Medicine, held in Lviv, Ukraine, in November, 2018. It comprises 30 contributed papers that were carefully peer-reviewed and selected from 53 submissions.

Copyright © 2018 for the individual papers by the papers' authors.  
Copying permitted only for private and academic purposes. This  
volume is published and copyrighted by its editors.

## **IDDM 2018**

# **Informatics & Data-Driven Medicine**

**Proceedings of the 1st International Workshop on Informatics & Data-Driven Medicine (IDDM 2018)**

**Lviv, Ukraine, November 28-30, 2018.**

**Edited by**

**Nataliya Shakhovska \***

**Sergio Montenegro \*\***

**Yannick Estève \*\*\***

**Sergey Subbotin \*\*\*\***

**Natalia Kryvinska \*\*\*\*\***

**Ivan Izonin \*\*\*\*\***

\* [Lviv Polytechnic National University, Ukraine](#)

\*\* [Julius-Maximilians-Universität Würzburg, Germany](#)

\*\*\* [University of Le Mans, France](#)

\*\*\*\* [Zaporozhye National Technical University, Ukraine](#)

\*\*\*\*\* [University of Vienna, Austria](#)

\*\*\*\*\* [Lviv Polytechnic National University, Ukraine](#)

## Table of Contents

- Cover and Preface
- Organization and Committees
- Organizers and Partners

### Session 1. Computational intelligence in medicine

- Synthesis of Artificial Neural Networks Using a Modified Genetic Algorithm 1-13  
*Serhii Leoshchenko, Andrii Oliinyk, Sergey Subbotin, Nataliia Gorobii, Tetiana Zaiko*
- Solving the Tension/Compression Spring Design Problem by an Improved Firefly Algorithm 14-20  
*Yuksel Celik, Hakan Kutucu*
- Fuzzy Clustering of Biomedical Datasets Using BSB-Neuro-Fuzzy-Model 21-28  
*Iryna Perova, Yevgeniy Bodyanskiy, Anatoliy Sachenko, Mikolaj Karpinski, Pawel Rudyk*
- Electroencephalogram Signals Analysis by Fuzzy Classifiers based on Cumulative Mutual Information 29-44  
*Jan Rabcan, Elena Zaitseva, Vitaly Levashenko*
- The use of Hybrid Computational Methods for Creating Intelligent Decision-Making Systems in Medicine 45-55  
*Pavel Klachek, Irina Liberman*
- Application of Cluster Analysis for Determination of the Oxidative Stress Decompensation Criteria 56-64  
*Myroslav Stupnytskyi, Viktor Zhukov, Oleksii Biletskii, Michal Gregus*
- Mesh Network of eHealth Intelligent Agents in Smart City: A Case Study on Assistive Devices for Visually Impaired People 65-81  
*Dmytro Zubov, Utku Kose, Ali Jasim Ramadhan, Andrey Kupin*
- The Sequential Associative Rules Analysis of Patient's Physical Characteristics 82-92  
*Nataliya Shakhovska, Roman Holoshchuk, Solomia Fedushko, Oleh Kosar, Roman Danel, Michal Řepka*
- Improving Automatic Categorization of Technical vs. Laymen Medical Words using FastText Word Embeddings 93-102  
*Hanna Pylieva, Artem Chernodub, Natalia Grabar, Thierry Hamon*
- The Personalized Approach to the Processing and Analysis of Patients' Medical Data 103-112  
*Nataliia Melnykova, Volodymyr Melnykov, Edgars Vasilevskis*
- Improving the Input Information for Medical Software Requirements Specifications using Ontology-Based Intelligent Agent 113-125  
*Tetiana Hovorushchenko, Olga Pavlova, Mykola Fedula*
- The Axiomatic-Deductive Strategy of Knowledge Organization in Onto-based e-learning Systems for Chinese Image Medicine 126-134  
*Serhii Lupenko, Volodymyr Pasichnyk, Natalia Kunanets, Oleksandra Orobchuk, Mingtang Xu*

- The Portraits Creating Method by Correlation Analysis of Hormone-Producing Cells Data 135-145  
*Olha Ryabukha, Ivanna Dronyuk*
- Complex Term Identification for Ukrainian Medical Texts 146-154  
*Olga Cherednichenko, Nadiia Babkova, Olga Kanishcheva*

## Session 2. High dimensional approximation and statistical learning

- The Regression Tree Model Building Based on a Cluster-Regression Approximation for Data-Driven Medicine 155-169  
*Sergey Subbotin, Elena Kirsanova*
- Piecewise-linear Approach for Medical Insurance Costs Prediction using SGTM Neural-Like Structure 170-179  
*Roman Tkachenko, Ivan Izonin, Natalia Kryvinska, Valentyna Chopyak, Nataliia Lotoshynska, Dmytro Danylyuk*
- Numerical Modeling and Analysis of Physical Properties in Biomaterials with Fractal Structure 180-192  
*Ivan Sokolovskyy, Maryana Levkovich, Olha Mokrytska*
- Method of Traffic Monitoring for DDoS Attacks Detection in e-Health Systems and Networks 193-204  
*Maksym Zalisky, Roman Odarchenko, Sergiy Gnatyuk, Yuliia Petrova, Anastasiia Chaplits*
- The Smart House Based System for the Collection and Analysis of Medical Data 205-214  
*Yurii Kryvenchuky, Nataliya Shakhovska, Iryna Shvorob, Sergio Montenegro, Maksym Nechepurenko*
- Mathematical Models of Biophysical Processes Taking Into Account Memory Effects and Self-Similarity 215-228  
*Yaroslav Sokolovskyy, Maryana Levkovich, Olha Mokrytska, Yaroslav Kaplunskyy*

## Session 3. Information systems in medicine: technology and applications

- Important Aspects to Consider When Developing ICTs for Purposes of Fall Prevention in the eHealth Domain 229-238  
*Jaime Campos, Linda Askenäs*
- The Consolidated Information Web-Resource about Pharmacy Networks in City 239-255  
*Victoria Vysotska, Vasyl Lytvyn, Yevhen Burov, Aleksandr Gozhij, Stepan Makara*
- Telecommunication Infrastructures for Telemedicine in Smart Cities 256-266  
*Volodymyr Pasichnyk, Nataliia Kunanets, Serhii Martsenko, Oleksandr Matsiuk, Olesia Mytnyk, Oleksii Duda, Pawel Falat*
- Information Technology of Traffic Modeling Using the Ateb-Functions Theory for Improving Service Delivery in eHealth Systems and Networks 267-274  
*Wingfrid Auzinger, Ivanna Dronyuk, Olga Fedevych, Yurii Klishch, Piotr Lipinski*
- Mobile Application for Preliminary Diagnosis of Diseases 275-286  
*Edgars Vasilevskis, Iryna Dubyak, Taras Basyuk, Volodymyr Pasichnyk, Antonii Rzhеuskyi*

- Information Technology Platform "Dental laboratory" 287-300  
*Yaroslav Vykylyuk, Nataliia Kunanets, Tetyana Kalahurka, Mykhailo Voronovsky*
- Verifying the Medical Specialty from User Profile of Online Community for Health-Related Advices 301-310  
*Solomia Fedushko, Natalia Shakhovska, Yuriy Syerov*

#### **Session 4. Biomedical image analysis and understanding**

- Modern Automated Microscopy Systems in Oncology 311-325  
*Oleh Berezsky, Oleh Pitsun, Natalia Batryn, Tamara Datsko, Kateryna Berezska, Lesia Dubchak*
- The Main Principles of Monitoring of Recurrent Laryngeal Nerve Monitoring During Surgery on Neck Organs 326-335  
*Mykola Dyvak, Volodymyr Tymets, Viktor Shidlovsky*
- The Method of Intelligent Image Processing Based on a Three-Channel Purely Convolutional Neural Network 336-351  
*Eugene Fedorov, Valentyna Lukashenko, Volodymyr Patrushev, Andriy Lukashenko, Kostiantyn Rudakov, Serhii Mitsenko*

---

2018-11-25: submitted by Ivan Izonin, metadata incl. bibliographic data published under [Creative Commons CC0](#)

2018-11-26: published on CEUR-WS.org |[valid HTML5](#)|

# The Method of Intelligent Image Processing Based on a Three-Channel Purely Convolutional Neural Network

Eugene Fedorov<sup>1</sup>[0000-0003-3841-7373], Valentyna Lukashenko<sup>1</sup>[0000-0002-6749-9040],  
Volodymyr Patrushev<sup>2</sup>[0000-0002-4061-4424], Andriy Lukashenko<sup>3</sup>[0000-0002-6016-1819],  
Kostiantyn Rudakov<sup>1</sup>[0000-0003-0000-6077], and Serhii Mitsenko<sup>1</sup>[0000-0002-9582-7486]

<sup>1</sup> Cherkasy State Technological University, Cherkasy, Shevchenko blvd., 460, 18006, Ukraine  
{ckc, k.rudakov, s.mitsenko}@chdtu.edu.ua, fedorovee75@ukr.net

<sup>2</sup> Donetsk National Technical University, Pokrovsk, Shybankova sq., 2, 85300, Ukraine  
wa\_pat@ukr.net

<sup>3</sup> Institute of Electric Welding E. O. Paton, Kyiv, Bozhenko str., 11, 03680, Ukraine  
ineks-kiev@ukr.net

**Abstract.** In the paper a method of intelligent image processing have been developed. This method based on a three channel purely convolutional neural network. The method consists of neural network model, a criterion to evaluate the effectiveness of the proposed model, a method for neural network learning in batch mode and a corresponding learning algorithm. This algorithm is intended for implementation on GPU by means of CUDA technology. The created model of neural network does not require the determination of the number of planes in hidden layers. This feature of the model simplifies its parametric identification "in large" and provides the use of three planes in the input layer. This simplifies the work with RGB images. The proposed method of intelligent image processing can be used in various intelligent systems of medical diagnostics.

**Keywords:** three-channel purely convolutional neural network, image recognition, batch learning mode, medical diagnostics, CUDA technologies.

## 1 Introduction

Currently, methods of automatic detection of mammary gland microcalcifications [1-2], nodes in the lungs [3]; polyps [4], pulmonary embolism [5]; brain tumors [6], etc., which are based on the approaches of artificial intelligence and are applied to digital images, are widely disseminated.

For image recognition, such approaches as [7]:

- metric [8, 9], which uses the metric to match the recognized and reference images;
- Bayesian [10], which uses a posteriori probability to match the recognized and reference images;
- generative [11-13], which uses a combination of a state machine and dynamic programming;

— neural [14-16], which uses an artificial neural network, are commonly used.

The highest probability of image recognition is achieved by means of deep neural networks.

Currently, the following deep neural networks are commonly used for image recognition:

- convolutional neural network (CNN) [17-18], which is a dynamic network;
- deep Boltzmann machine (DBM) [19-20], which is a recurrent network;
- deep autoencoder [21], which is a static network;
- neocognitron [22], which is a dynamic network.

Compared with other deep neural networks, CNN has two advantages at the same time – the possibility of a batch learning mode and the highest probability of recognition.

The disadvantages of traditional CNN include the lack of binding of its learning procedure to a parallel architecture, the lack of automatic determination of the number of planes in its hidden layers, the consideration of its learning procedure for only one plane of the input layer. In this regard, it is relevant to create a modified CNN, which will eliminate these drawbacks.

The goal of the work is to create a method of intelligent image processing based on three-channel purely convolutional neural network. To achieve the goal, the following tasks have been set and solved:

1. to create a model of three-channel purely convolutional neural network;
2. to choose a criterion for evaluating the effectiveness of the proposed model;  
to develop a method for learning a three-channel purely convolutional neural network in batch mode;
3. to create a learning algorithm for three-channel purely convolutional neural network in batch mode, designed for implementation on GPU by means of CUDA technology;
4. to conduct a numerical study.

## **2 The creation of a model of three-channel purely convolutional neural network**

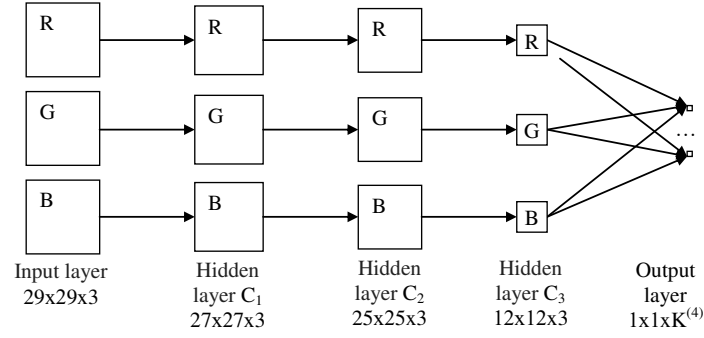
In this paper a modification of CNN – three-channel purely convolutional neural network (3PCNN) – is offered. An example of 3PCNN for a feature matrix of  $29 \times 29$  in size for three color components is shown in Fig. 1. Geometrically, communication area for the input, first, and second convolutional layers is a  $3 \times 3$  square.

Unlike usual CNN, the proposed 3PCNN has the following features:

- there are three consecutive convolutional layers;
- the third convolutional layer replaces subsampling layer (communication area is shifted not by 1, but by  $q$ );
- the number of planes for the input and convolutional layers is three;



- each plane of the input layer is associated with only one plane of the first convolutional layer;
- each plane of the first convolutional layer is associated with only one plane of the second convolutional layer;
- each plane of the second convolutional layer is associated with only one plane of the third convolutional layer.



**Fig. 1.** Three-channel purely convolutional neural network (3PCNN).

The 3PCNN model is presented as follows:

$$u^{(0)}(m, k) = x(m, k), m \in \{1, \dots, N/2\}^2, k \in \overline{1, 3},$$

$$u^{(1)}(m, k) = f^{(1)}(h^{(1)}(m, k)), m \in \{1, \dots, N^{(1)}\}^2, k \in \overline{1, 3},$$

$$h^{(1)}(m, k) = b^{(1)}(k) + \sum_v w^{(1)}(v, k, k) u^{(0)}(m+v, k), v \in V_m^{(0)},$$

$$u^{(2)}(m, k) = f^{(2)}(h^{(2)}(m, k)), m \in \{1, \dots, N^{(2)}\}^2, k \in \overline{1, 3},$$

$$h^{(2)}(m, k) = b^{(2)}(k) + \sum_v w^{(2)}(v, k, k) u^{(1)}(m+v, k), v \in V_m^{(1)},$$

$$u^{(3)}(m, k) = f^{(3)}(h^{(3)}(m, k)), m \in \{1, \dots, N^{(3)}\}^2, k \in \overline{1, 3},$$

$$h^{(3)}(m, k) = b^{(3)}(k) + \sum_v w^{(3)}(v, k, k) u^{(2)}(qm+v-(1-q, 1-q), k), v \in V_m^{(2)},$$

$$u^{(4)}(i) = f^{(4)}(h^{(4)}(i)), i \in \overline{1, K^{(4)}},$$

$$h^{(4)}(i) = b^{(4)}(i) + \sum_{k=1}^3 \sum_m w^{(4)}(m, k, i) u^{(3)}(m, k), m \in \{1, \dots, N^{(3)}\}^2,$$

where  $i$  – the number of the plane of output layer cells,

$k$  – the number of the plane of convolutional layer cells  $C_1, C_2, C_3$ ,  
 $q$  – subsampling coefficient, which is a natural number,  
 $\nu$  – position in communication area,  $\nu = (\nu_x, \nu_y)$ ,  
 $V_m^{(l)}$  – communication area of the  $l^{\text{th}}$  layer for the neuron in  $m$  position of the  $l+1^{\text{st}}$  layer,  
 $K^{(4)}$  – the number of cell planes in the output layer,  
 $b^{(1)}(k)$  – threshold values for the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_1$  layer,  
 $b^{(2)}(k)$  – threshold values for the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_2$  layer,  
 $b^{(3)}(k)$  – threshold values for the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_3$  layer,  
 $b^{(4)}(i)$  – threshold values for the neuron of the  $i^{\text{th}}$  plane of the output layer,  
 $w^{(1)}(\nu, k, k)$  – connection weight from the neuron in  $m+\nu$  position of the  $k^{\text{th}}$  plane of the input layer to the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_1$  layer,  
 $w^{(2)}(\nu, k, k)$  – connection weight from the neuron in  $m+\nu$  position of the  $k^{\text{th}}$  plane of  $C_1$  layer to the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_2$  layer,  
 $w^{(3)}(\nu, k, k)$  – connection weight from the neuron in  $qm+\nu-(1-q, 1-q)$  position of the  $k^{\text{th}}$  plane of  $C_2$  layer to the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_3$  layer,  
 $w^{(4)}(m, k, i)$  – connection weight from the neuron in  $m$  position of the  $k^{\text{th}}$  plane of  $C_3$  layer to the neuron of the  $i^{\text{th}}$  plane of the output layer,  
 $u^{(1)}(m, k)$  – neuron yield in  $m$  position of the  $k^{\text{th}}$  plane of  $C_1$  layer,  
 $u^{(2)}(m, k)$  – neuron yield in  $m$  position of the  $k^{\text{th}}$  plane of  $C_2$  layer,  
 $u^{(3)}(m, k)$  – neuron yield in  $m$  position of the  $k^{\text{th}}$  plane of  $C_3$  layer,  
 $u^{(4)}(i)$  – neuron yield of the  $i^{\text{th}}$  plane of the output layer,  
 $f^{(1)}$  – the ReLU function of neurons activation in  $C_1$  layer,  
 $f^{(2)}$  – the ReLU function of neurons activation in  $C_2$  layer,  
 $f^{(3)}$  – the ReLU function of neurons activation in  $C_3$  layer,  
 $f^{(4)}$  – the logistic function of neurons activation in the output layer.

### 3 The choice of the criterion for evaluation of the proposed model effectiveness

In the work, for 3PCNN model learning the goal function is chosen, that means the choice of such parameter values, which deliver the minimum of root-mean-square error (the difference of model output and test output):

$$F = \frac{1}{PK^{(4)}} \sum_{\mu=1}^P \sum_{i=1}^{K^{(4)}} (u_{\mu}^{(4)}(i) - d_{\mu i})^2 \rightarrow \min_W \quad (1)$$

where  $u_{\mu}^{(4)} = (u_{\mu}^{(4)}(1), \dots, u_{\mu}^{(4)}(K^{(4)}))$  – the  $\mu$ <sup>th</sup> model output signal,

$d_{\mu} = (d_{\mu 1}, \dots, d_{\mu K^{(4)}})$  – the  $\mu$ <sup>th</sup> test output signal,

$P$  – test set power,

$W$  – 3PCNN parameters,  $W = \{w^{(1)}(v, k, k), w^{(2)}(v, k, k), w^{(3)}(k, k), w^{(4)}(m, k, i)\}$ .

#### 4 The development of learning method for three-channel purely convolutional neural network in batch mode

In this paper, for 3PCNN an error correction learning (teacher learning), using the back propagation method (BP), is used. This is an iterative gradient learning method which minimizes root-mean-square error. In this paper, batch learning mode for parallel computing is offered.

The structure of the method for 3PCNN learning in batch mode is presented in Fig. 2.

The method for 3PCNN learning in batch mode consists of the following blocks.

##### Block 1 – initialize 3PCNN parameters

Set the number of the current learning era  $n$  by one.

$$b^{(1)}(n, k) = rand(), \quad k \in \overline{1, 3},$$

$$w^{(1)}(n, v, k, k) = rand(), \quad v \in V^{(0)}, \quad k \in \overline{1, 3},$$

$$b^{(2)}(n, k) = rand(), \quad k \in \overline{1, 3},$$

$$w^{(2)}(n, v, k, k) = rand(), \quad v \in V^{(1)}, \quad k \in \overline{1, 3},$$

$$b^{(3)}(n, k) = rand(), \quad k \in \overline{1, 3},$$

$$w^{(3)}(n, v, k, k) = rand(), \quad v \in V^{(2)}, \quad k \in \overline{1, 3},$$

$$b^{(4)}(n, i) = rand(), \quad i \in \overline{1, K^{(4)}},$$

$$w^{(4)}(n, m, k, i) = rand(), \quad m \in \{1, \dots, N^{(3)}\}^2, \quad k \in \overline{1, 3}, \quad i \in \overline{1, K^{(4)}},$$

where  $V^{(l)}$  – communication area in the  $l$ <sup>th</sup> layer.

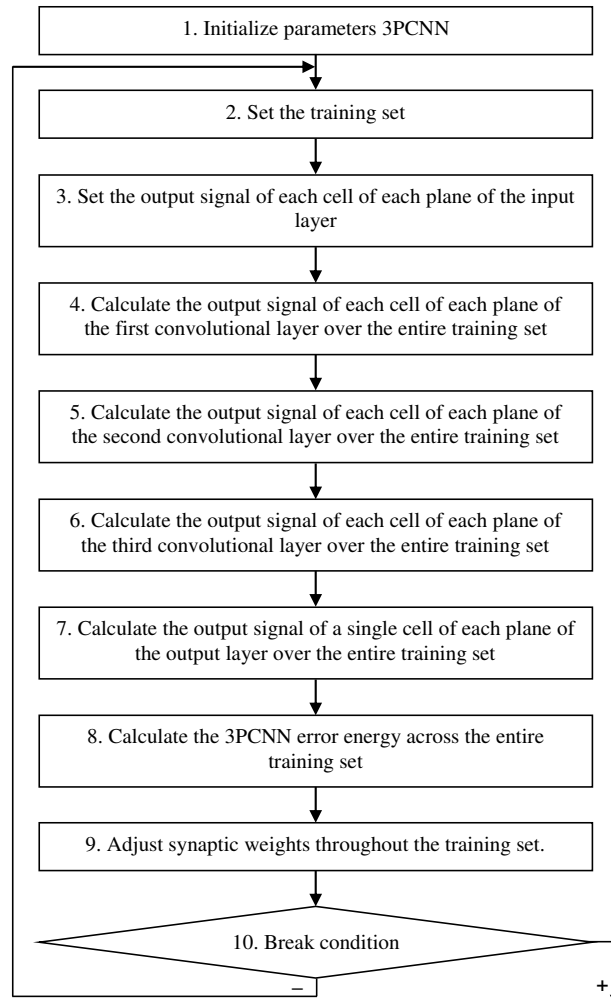
**Block 2 – Specify the learning set**

$$\{(\mathbf{x}_\mu, \mathbf{d}_\mu) \mid \mathbf{x}_\mu \in R^{(N/2) \times (N/2)}, \mathbf{d}_\mu \in \{0,1\}^{K^{(4)}}, \mu \in \overline{1, P},$$

where  $\mathbf{x}_\mu$  – the  $\mu^{\text{th}}$  learning input matrix of features,

$\mathbf{d}_\mu$  – the  $\mu^{\text{th}}$  learning output vector,

$P$  – learning set power.



**Fig. 2.** The structure of the method for 3PCNN learning in batch mode.

**Block 3 – Set the output signal of each cell of each plane of the input layer**

$$u_\mu^{(0)}(n, m, k) = x_\mu(m, k), \quad m \in \{1, \dots, N/2\}^2, \quad k \in \overline{1, 3}, \quad \mu \in \overline{1, P}.$$

**Block 4 – Calculate the output signal of each cell of each plane of the first convolutional layer over the entire learning set**

$$u_{\mu}^{(1)}(n, m, k) = f^{(1)}(h_{\mu}^{(1)}(n, m, k)), \quad m \in \{1, \dots, N^{(1)}\}^2, \quad k \in \overline{1, 3}, \quad \mu \in \overline{1, P},$$

$$h_{\mu}^{(1)}(n, m, k) = b^{(1)}(n, k) + \sum_{\nu} w^{(1)}(n, \nu, k, k) u_{\mu}^{(0)}(n, m + \nu, k), \quad \nu \in V_m^{(0)}.$$

**Block 5 – Calculate the output signal of each cell of each plane of the second convolutional layer over the entire learning set**

$$u_{\mu}^{(2)}(n, m, k) = f^{(2)}(h_{\mu}^{(2)}(n, m, k)), \quad m \in \{1, \dots, N^{(2)}\}^2, \quad k \in \overline{1, 3}, \quad \mu \in \overline{1, P},$$

$$h_{\mu}^{(2)}(n, m, k) = b^{(2)}(n, k) + \sum_{\nu} w^{(2)}(n, \nu, k, k) u_{\mu}^{(1)}(n, m + \nu, k), \quad \nu \in V_m^{(1)}.$$

**Block 6 – Calculate the output signal of each cell of each plane of the third convolutional layer over the entire learning set**

$$u_{\mu}^{(3)}(n, m, k) = f^{(3)}(h_{\mu}^{(3)}(n, m, k)), \quad m \in \{1, \dots, N^{(3)}\}^2, \quad k \in \overline{1, 3}, \quad \mu \in \overline{1, P},$$

$$h_{\mu}^{(3)}(n, m, k) = b^{(3)}(n, k) + \sum_{\nu} w^{(3)}(n, \nu, k, k) u^{(2)}(n, qm + \nu - (1 - q, 1 - q), k), \quad \nu \in V_m^{(2)}.$$

**Block 7 – Calculate the output signal of a single cell of each plane of the output layer over the entire learning set**

$$u_{\mu}^{(4)}(n, i) = f^{(4)}(h_{\mu}^{(4)}(n, i)), \quad i \in \overline{1, K^{(4)}}, \quad \mu \in \overline{1, P},$$

$$h_{\mu}^{(4)}(n, i) = b^{(4)}(n, i) + \sum_{k=1}^3 \sum_m w^{(4)}(n, m, k, i) u_{\mu}^{(3)}(n, m, k), \quad m \in \{1, \dots, N^{(3)}\}^2.$$

**Block 8 – Calculate the energy of 3PCNN error over the entire learning set**

$$E(n) = \frac{1}{2P} \sum_{\mu=1}^P \sum_{i=1}^{K^{(4)}} e_{\mu i}^2(n), \quad e_{\mu i}(n) = u_{\mu}^{(4)}(n, i) - d_{\mu i}.$$

**Block 9 – Adjust synaptic weights based on generalized delta rule over the entire learning set**

$$b^{(4)}(n+1, i) = b^{(4)}(n, i) - \eta(n) \frac{\partial E(n)}{\partial b^{(4)}(n, i)}, \quad i \in \overline{1, K^{(4)}},$$

$$w^{(4)}(n+1, m, k, i) = w^{(4)}(n, m, k, i) - \eta(n) \frac{\partial E(n)}{\partial w^{(4)}(n, m, k, i)}, \quad m \in \{1, \dots, N^{(3)}\}^2, \quad k \in \overline{1, 3}, \quad i \in \overline{1, K^{(4)}},$$

$$b^{(3)}(n+1, k) = b^{(3)}(n, k) - \eta(n) \frac{\partial E(n)}{\partial b^{(3)}(n, k)}, \quad k \in \overline{1, 3},$$

$$w^{(3)}(n+1, \nu, k, k) = w^{(3)}(n, \nu, k, k) - \eta(n) \frac{\partial E(n)}{\partial w^{(3)}(n, \nu, k, k)}, \quad \nu \in V^{(2)}, \quad k \in \overline{1, 3},$$

$$b^{(2)}(n+1, k) = b^{(2)}(n, k) - \eta(n) \frac{\partial E(n)}{\partial b^{(2)}(n, k)}, \quad k \in \overline{1, 3},$$

$$w^{(2)}(n+1, \nu, k, k) = w^{(2)}(n, \nu, k, k) - \eta(n) \frac{\partial E(n)}{\partial w^{(2)}(n, \nu, k, k)}, \quad \nu \in V^{(1)}, \quad k \in \overline{1, 3},$$

$$b^{(1)}(n+1, k) = b^{(1)}(n, k) - \eta(n) \frac{\partial E(n)}{\partial b^{(1)}(n, k)}, \quad k \in \overline{1, 3},$$

$$w^{(1)}(n+1, \nu, k, k) = w^{(1)}(n, \nu, k, k) - \eta(n) \frac{\partial E(n)}{\partial w^{(1)}(n, \nu, k, k)}, \quad \nu \in V^{(0)}, \quad k \in \overline{1, 3},$$

$$\frac{\partial E(n)}{\partial b^{(4)}(n, i)} = \frac{1}{P} \sum_{\mu=1}^P g_{\mu}^{(4)}(n, i),$$

$$\frac{\partial E(n)}{\partial w^{(4)}(n, m, k, i)} = \frac{1}{P} \sum_{\mu=1}^P u_{\mu}^{(3)}(n, m, k) g_{\mu}^{(4)}(n, i),$$

$$\frac{\partial E(n)}{\partial b^{(3)}(n, k)} = \frac{1}{P} \sum_{\mu=1}^P \sum_m g_{\mu}^{(3)}(n, m, k), \quad m \in \{1, \dots, N^{(2)}\}^2,$$

$$\frac{\partial E(n)}{\partial w^{(3)}(n, \nu, k, k)} = \frac{1}{P} \sum_{\mu=1}^P \sum_m u_{\mu}^{(2)}(n, qm + \nu + (1-q, 1-q), k) g_{\mu}^{(3)}(n, m, k), \quad m \in \{1, \dots, N^{(2)}\}^2,$$

$$\frac{\partial E(n)}{\partial b^{(2)}(n, k)} = \frac{1}{P} \sum_{\mu=1}^P \sum_m g_{\mu}^{(2)}(n, m, k), \quad m \in \{1, \dots, N^{(1)}\}^2,$$

$$\frac{\partial E(n)}{\partial w^{(2)}(n, \nu, k, k)} = \frac{1}{P} \sum_{\mu=1}^P \sum_m u_{\mu}^{(1)}(n, m + \nu, k) g_{\mu}^{(2)}(n, m, k), \quad m \in \{1, \dots, N^{(1)}\}^2,$$

$$\frac{\partial E(n)}{\partial b^{(1)}(n, k)} = \frac{1}{P} \sum_{\mu=1}^P \sum_m g_{\mu}^{(1)}(n, m, k), \quad m \in \{1, \dots, N/2\}^2,$$

$$\frac{\partial E(n)}{\partial w^{(1)}(n, \nu, k, k)} = \frac{1}{P} \sum_{\mu=1}^P \sum_m u_{\mu}^{(0)}(n, m + \nu, k) g_{\mu}^{(1)}(n, m, k), \quad m \in \{1, \dots, N/2\}^2,$$

$$g_{\mu}^{(4)}(n, i) = e_{\mu i}(n) f^{(4)}(h_{\mu}^{(4)}(n, i)),$$

$$g_{\mu}^{(3)}(n, m, k) = f^{(3)}(h_{\mu}^{(3)}(n, m, k)) \sum_{i=1}^{K^{(4)}} w^{(4)}(n, m, k, i) g_{\mu}^{(4)}(n, i),$$

$$g_{\mu}^{(2)}(n, m, k) = f^{(2)}(h_{\mu}^{(2)}(n, m, k)) \cdot$$

$$\sum_{\nu} w^{(3)}(n, \nu, k, k) g_{\mu}^{(3)} \left( n, \left[ \frac{m - \nu - (1, 1)}{q} \right] + (1, 1), k \right), \quad \nu \in D_m^{(3)},$$

$$g_{\mu}^{(1)}(n, m, k) = f^{(1)}(h_{\mu}^{(1)}(n, m, k)) \sum_{\nu} w^{(2)}(n, \nu, k, k) g_{\mu}^{(2)}(n, m - \nu, k), \quad \nu \in D_m^{(2)},$$

where  $D_m^{(l)}$  – communication area of the  $l^{\text{th}}$  layer for the neuron in  $m$  position of the  $l-1^{\text{th}}$  layer,

$\eta(n)$  – parameter that determines the learning rate in accordance with the rule of exponential decay,

$$\eta(n) = \eta(0) e^{-\text{decay\_rate} \cdot t}; \quad \text{decay\_rate} > 0,$$

where  $\eta(0)$  – initial learning rate,  $\eta(0) > 0$ .

#### **Block 10 – Check termination condition.**

If  $E(n) \geq \varepsilon$ , then increase the number of the era  $n$  by one and go to block 2.

## **5 The creation of learning algorithm of three-channel purely convolutional neural network in batch mode**

The algorithm for 3PCNN learning in batch mode, designed for implementation on GPU by means of CUDA technology, is shown in Fig. 3. This block diagram functions as follows.

Step 1 – Download the initial values of 3PCNN parameters into GPU global memory.

Step 2 – Download the learning set into GPU global memory.

Step 3 – Download the output signal of each cell of each plane of the input layer into GPU global memory.

Step 4 – Calculate the output signal of each cell of each plane of the first convolutional layer  $u_{\mu}^{(1)}(n, m, k)$  over the entire learning set using  $3P(N^{(1)})^2$  threads which

are grouped into  $(3P(N^{(1)})^2)/N^s$  blocks, where  $N^s$  – the number of threads in the block. Each thread computes

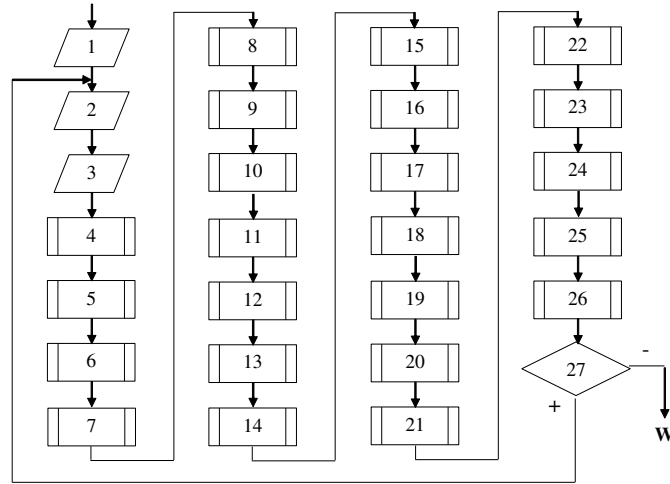
$$u_{\mu}^{(1)}(n, m, k) = f^{(1)}(h_{\mu}^{(1)}(n, m, k)),$$

$$h_{\mu}^{(1)}(n, m, k) = b^{(1)}(n, k) + \sum_{\nu} w^{(1)}(n, \nu, k, k) u_{\mu}^{(0)}(n, m + \nu, k), \quad \nu \in V_m^{(0)}.$$

Step 5 – Calculate the output signal of each cell of each plane of the second convolutional layer  $u_{\mu}^{(2)}(n, m, k)$  over the entire learning set using  $3P(N^{(2)})^2$  threads which are grouped into  $(3P(N^{(2)})^2)/N^s$  blocks. Each thread computes

$$u_{\mu}^{(2)}(n, m, k) = f^{(2)}(h_{\mu}^{(2)}(n, m, k)),$$

$$h_{\mu}^{(2)}(n, m, k) = b^{(2)}(n, k) + \sum_{\nu} w^{(2)}(n, \nu, k, k) u_{\mu}^{(1)}(n, m + \nu, k), \quad \nu \in V_m^{(0)}.$$



**Fig. 3.** Block diagram of the algorithm for 3PCNN learning in batch mode.

Step 6 – Calculate the output signal of each cell of each plane of the third convolutional layer  $u_{\mu}^{(3)}(n, m, k)$  over the entire learning set using  $3P(N^{(2)})^2$  threads which are grouped into  $(3P(N^{(2)})^2)/N^s$  blocks. Each thread computes

$$u_{\mu}^{(3)}(n, m, k) = f^{(3)}(h_{\mu}^{(3)}(n, m, k)),$$

$$h_{\mu}^{(3)}(n, m, k) = b^{(3)}(n, k) + \sum_{\nu} w^{(3)}(n, \nu, k, k) u_{\mu}^{(2)}(n, qm + \nu - (1 - q, 1 - q), k), \quad \nu \in V_m^{(2)}.$$



Step 7 – Calculate the  $s_\mu(n,i)$  sum over the entire learning set using  $3P(N^{(3)})^2 K^{(4)}$  threads which are grouped into  $\left(3P(N^{(3)})^2 K^{(4)}\right)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $w^{(4)}(n,m,k,i)u_\mu^{(3)}(n,m,k)$  form is calculated. Partial sums, received in each block, are added up, based on the reduction.

Step 8 – Calculate the output signal of a single cell of each plane of the output  $u_\mu^{(4)}(n,i)$  layer over the entire learning set using  $PK^{(4)}$  threads which are grouped into  $\left(PK^{(4)}\right)/N^s$  blocks. Each thread computes

$$u_\mu^{(4)}(n,i) = f^{(4)}(h_\mu^{(4)}(n,i)),$$

$$h_\mu^{(4)}(n,i) = b^{(4)}(n,i) + s_\mu(n,i).$$

Step 9 – Calculate the energy of error  $E(n)$  over the entire learning set using  $PK^{(4)}$  threads which are grouped into  $\left(PK^{(4)}\right)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $u_\mu^{(4)}(n,i) - d_{\mu i}$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $2P$ .

Step 10 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial b^{(4)}(n,i)}$  over the entire learning set using  $PK^{(4)}$  threads which are grouped into  $\left(PK^{(4)}\right)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $g_\mu^{(4)}(n,i) = e_{\mu i}(n) f'^{(4)}(h_\mu^{(4)}(n,i))$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 11 – Calculate the threshold value  $b^{(4)}(n+1,i)$  using  $K^{(4)}$  threads which are grouped into one block. Each thread computes

$$b^{(4)}(n+1,i) = b^{(4)}(n,i) - \eta(n) \frac{\partial E(n)}{\partial b^{(4)}(n,i)}.$$

Step 12 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial w^{(4)}(n,m,k,i)}$  over the entire learning set using  $3P(N^{(3)})^2 K^{(4)}$  threads which are grouped into  $\left(3P(N^{(3)})^2 K^{(4)}\right)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $u_\mu^{(3)}(n,m,k)g_\mu^{(4)}(n,i)$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 13 – Calculate synaptic weight  $w^{(4)}(n+1, m, k, i)$  using  $3(N^{(3)})^2 K^{(4)}$  threads which are grouped into  $\left(3(N^{(3)})^2 K^{(4)}\right)/N^s$  blocks. Each thread computes

$$w^{(4)}(n+1, m, k, i) = w^{(4)}(n, m, k, i) - \eta(n) \frac{\partial E(n)}{\partial w^{(4)}(n, m, k, i)}.$$

Step 14 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial b^{(3)}(n, k)}$  over the entire learning set using  $3P(N^{(2)})^2$  threads which are grouped into  $\left(3P(N^{(2)})^2\right)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $g_\mu^{(3)}(n, m, k) = f^{(3)}(h_\mu^{(3)}(n, m, k)) \sum_{i=1}^{K^{(4)}} w^{(4)}(n, m, k, i) g_\mu^{(4)}(n, i)$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 15 – Calculate the threshold value  $b^{(3)}(n+1, k)$  using three threads, which are grouped into one block. Each thread computes

$$b^{(3)}(n+1, k) = b^{(3)}(n, k) - \eta(n) \frac{\partial E(n)}{\partial b^{(3)}(n, k)}.$$

Step 16 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial w^{(3)}(n, \nu, k, k)}$  over the entire learning set using  $3P(N^{(2)})^2 |V^{(2)}|$  threads which are grouped into  $\frac{3P(N^{(2)})^2 |V^{(2)}|}{N^s}$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $u_\mu^{(2)}(n, qm + \nu + (1-q, 1-q), k) g_\mu^{(3)}(n, m, k)$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 17 – Calculate synaptic weight  $w^{(3)}(n+1, \nu, k, k)$  using  $3|V^{(2)}|$  threads which are grouped into one block. Each thread computes

$$w^{(3)}(n+1, \nu, k, k) = w^{(3)}(n, \nu, k, k) - \eta(n) \frac{\partial E(n)}{\partial w^{(3)}(n, \nu, k, k)}.$$

Step 18 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial b^{(2)}(n, k)}$  over the entire learning set using  $3P(N^{(1)})^2$  threads which are grouped into  $\left(3P(N^{(1)})^2\right)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the

$$g_{\mu}^{(2)}(n, m, k) = f^{r(2)}(h_{\mu}^{(2)}(n, m, k)) \cdot \sum_{\nu} w^{(3)}(n, \nu, k, k) g_{\mu}^{(3)}\left(n, \left[\frac{m - \nu - (1, 1)}{q}\right] + (1, 1), k\right)$$

form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 19 – Calculate the threshold value  $b^{(2)}(n+1, k)$  using three threads, which are grouped into one block. Each thread computes

$$b^{(2)}(n+1, k) = b^{(2)}(n, k) - \eta(n) \frac{\partial E(n)}{\partial b^{(2)}(n, k)}.$$

Step 20 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial w^{(2)}(n, \nu, k, k)}$  over the entire learning set using  $3P(N^{(1)})^2 |V^{(1)}|$  threads which are grouped into  $\frac{3P(N^{(1)})^2 |V^{(1)}|}{N^s}$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $u_{\mu}^{(1)}(n, m + \nu, k) g_{\mu}^{(2)}(n, m, k)$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 21 – Calculate synaptic weight  $w^{(2)}(n+1, \nu, k, k)$  using  $3|V^{(1)}|$  threads which are grouped into one block. Each thread computes

$$w^{(2)}(n+1, \nu, k, k) = w^{(2)}(n, \nu, k, k) - \eta(n) \frac{\partial E(n)}{\partial w^{(2)}(n, \nu, k, k)}.$$

Step 22 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial b^{(1)}(n, k)}$  over the entire learning set using  $3P(N/2)^2$  threads which are grouped into  $(3P(N/2)^2)/N^s$  blocks. In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $g_{\mu}^{(1)}(n, m, k) = f^{r(1)}(h_{\mu}^{(1)}(n, m, k)) \sum_{\nu} w^{(2)}(n, \nu, k, k) g_{\mu}^{(2)}(n, m - \nu, k)$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 23 – Calculate the threshold value  $b^{(1)}(n+1, k)$  using three threads which are grouped into one block. Each thread computes

$$b^{(1)}(n+1, k) = b^{(1)}(n, k) - \eta(n) \frac{\partial E(n)}{\partial b^{(1)}(n, k)}.$$

Step 24 – Calculate the partial derivative  $\frac{\partial E(n)}{\partial w^{(1)}(n, \nu, k, k)}$  over the entire learning set using  $3P(N/2)^2 |V^{(0)}|$  threads which are grouped into  $\frac{3P(N/2)^2 |V^{(0)}|}{N^s}$  blocks.

In each block, based on the reduction, a partial sum from  $N^s$  elements of the  $u_{\mu}^{(0)}(n, m + \nu, k)g_{\mu}^{(1)}(n, m, k)$  form is calculated. Partial sums, received in each block, are added, based on the reduction, and the result is divided by  $P$ .

Step 25 – Calculate synaptic weight  $w^{(1)}(n+1, \nu, k, k)$  using  $3|V^{(0)}|$  threads which are grouped into one block. Each thread computes

$$w^{(1)}(n+1, \nu, k, k) = w^{(1)}(n, \nu, k, k) - \eta(n) \frac{\partial E(n)}{\partial w^{(1)}(n, \nu, k, k)}.$$

Step 26 – Calculate learning speed

$$\eta(n) = \eta(0)e^{-decay\_rate \cdot n}; \quad decay\_rate > 0.$$

Step 27 – Check termination condition.

If  $E(n) \geq \varepsilon$ , then increase the number of era  $n$  by one and go to step 2.

## 6 Numerical research

Recognition probabilities, obtained on the basis of CIFAR-10 based on traditional CNN and the proposed 3PCNN, are presented in Table 1. At the same time, traditional CNN had three stages (each consisted of a convolutional and subsampling layer with 5x5 and 2x2 communication areas, respectively), the first stage had four planes for each layer, the second stage had 16 planes for each layer, the third stage had 64 planes for each layer. The tests were carried out on GeForce 920M video card with the number of threads in the block  $N^s = 1024$ .

According to Table 1, 3PCNN with image preprocessing gives the best results.

**Table 1.** Image recognition probability.

Neural networks	Recognition probability
CNN	0.97
3PCNN	0.98

## Conclusions

1. To solve the problem of improving the quality of intelligent medical image processing, appropriate methods for image recognition have been investigated. These

studies have shown that the use of convolutional neural networks is currently the most effective.

2. The created model of three-channel purely convolutional neural network does not require the determination of the number of planes in hidden layers, that simplifies its parametric identification “in large” and ensures the use of three planes in the input layer, which simplifies the work with RGB images.
3. The created algorithm for learning a three-channel purely convolutional neural network is designed for software implementation on GPU by means of CUDA technology.
4. The proposed method of intelligent image processing, based on three-channel pure convolutional neural network, can be used in various intelligent systems of medical diagnostics.

## References

1. Zhang, W., Doi, K., Giger, M., Wu, Y., Nishikawa, R., Schmidt, R.: Computerized detection of clustered microcalcifications in digital mammograms using a shift-invariant artificial neural network. In: Williamson, J. (eds.) *Medical Physics*, vol. 21, pp. 517–524, (1994) doi.org/10.1118/1.597177
2. Chan, H., Lo, S., Lam, S., Helvie, M.: Computer-aided detection of mammographic microcalcifications: Pattern recognition with an artificial neural network. In: Williamson, J. (eds.) *Medical Physics*, vol. 22, pp. 1555–1567, (1995) doi.org/10.1118/1.597428
3. Lo, S., Lou, S., Lin, J., Freedman, M., Chien, M., Mun, S.: Artificial convolution neural network techniques and applications for lung nodule detection. In: Insana, M. (eds.) *Medical Imaging, IEEE Transactions*, vol. 14, pp. 711–718, (1995) doi.org/10.1109/42.476112
4. Tajbakhsh, N., Gurudu, S., Liang, J.: A comprehensive computer-aided polyp detection system for colonoscopy videos. In: Ourselin, S., Alexander, D., Westin, C., Cardoso, M. (eds.) *Information Processing in Medical Imaging*. Springer, pp. 327–338, (2015) doi.org/10.1007/978-3-319-19992-4\_25
5. Tajbakhsh, N., Liang, J.: Computer-aided pulmonary embolism detection using a novel vessel-aligned multi-planar image representation and convolutional neural networks. In: Navab, N., (eds.) *MICCAI 2015, Part II, LNCS 9350*, pp. 62–69 (2015) doi.org/10.1007/978-3-319-24571-3\_8
6. Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., Pal C., Jodoin, P, Larochelle, H.: Brain tumor segmentation with deep neural networks. In: *Medical Image Analysis*, vol. 35, pp. 18–31, (2017) doi.org/10.1016/j.media.2016.05.004
7. Fedorov E., Tossoriteit E.: *Models and methods of spectator images recognition*. Donetsk: Knowledge (Donetsk branch), 422 pp. (2013) in Russian
8. Brunelli, R., Poggio, T.: Face recognition: features versus templates. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, pp. 235–241, (1993) doi.org/10.1109/34.254061

9. Baron, R.: Mechanisms of human facial recognition. In: *International Journal of Man-Machine Studies*, vol. 15, pp. 137–178, (2008) doi.org/10.1016/S0020-7373(81)80001-6
10. Moghaddam, B., Jebara, T., Pentland, A.: Bayesian Face Recognition. In: *Pattern recognition*, vol. 33, pp. 1771–1782, (2000) doi.org/10.1016/S0031-3203(99)00179-X
11. Nefian, A., Hayes, M.: Hidden Markov models for face recognition. In: *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98*, pp. 2721–2724, (1998) doi.org/10.1109/ICASSP.1998.678085
12. Kohir, V., Desai, U.: Face recognition using DCT-HMM approach. In: *Proceedings Fourth IEEE Workshop on Applications of Computer Vision. WACV'98*, pp. 3400–3410, (1998) doi.org/10.1109/ACV.1998.732884
13. Daleno, D., Cariello, L., Giannini, M., Mastronardi, G.: Pseudo 2D hidden Markov model and neural network coefficients in face recognition. In: Oravec, M. (eds.) *Face Recognition*, pp. 151–170, (2010).
14. Tkachenko, R., Izonin, I.: Model and Principles for the Implementation of Neural-Like Structures based on Geometric Data Transformations. In: Hu, Z., Petoukhov, S., (eds) *Advances in Computer Science for Engineering and Education. ICCSEEA2018. Advances in Intelligent Systems and Computing. Springer, Cham*, vol.754, pp.578-587, (2018) doi.org/10.1007/978-3-319-91008-6\_58
15. Lyubchyk, L., Bodyansky, E., Rivtis, A.: Adaptive harmonic components detection and forecasting in wave non-periodic time series using neural networks. In: *ISCDMCI'2002. - Conf*, pp. 433-435, (2002).
16. Subbotin, S.: The special deep neural network for stationary signal spectra classification. In: *2018 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2018 – Proceedings*. (2018) doi.org/10.1109/TCSET.2018.8336170
17. LeCun, Y., Bengio, Y.: Convolutional networks for images, speech, and time series. In: Arbib, M. (eds.) *The handbook of brain theory and neural networks*, MIT Press Cambridge, pp. 255–258, (1998).
18. Lawrence, S., Giles, C., Tsoi, A., Back, A.: Face recognition: a convolutional neural network approach. In: *IEEE Transactions on Neural Networks*, vol. 8, pp. 98–113, (1997) doi.org/10.1109/72.554195
19. Salakhutdinov, R., Hinton, G.: Deep Boltzmann machines. *Journal of Machine Learning Research*, vol. 5, pp. 448–455, (2009).
20. Salakhutdinov, R., Larochelle, H.: Efficient learning of deep Boltzmann machines. In: *Journal of Machine Learning Research*, vol. 9, pp. 693–700, (2010).
21. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. *Science*, vol. 313, pp. 504–507, (2006) doi.org/10.1126/science.1127647
22. Fukushima, K.: Neocognitron for handwritten digit recognition. *Neurocomputing*, vol. 51, pp. 161–180, (2003) doi.org/10.1016/S0925-2312(02)00614-8