Artificial neural network for bubbles pattern recognition on the images

I E Poletaev[1,2], K S Pervunin[1,2], M P Tokarev[1]

[1] Kutateladze Institute of Thermophysics, the Russian Academy of Sciences, Siberian Branch
1, Lavrentyev Ave., Novosibirsk, 630090, Russia
[2]Novosibirsk National Research State University
2, Pirogova St., Novosibirsk, 630090, Russia

E-mail: poletaev.igor.phys@gmail.com

Abstract. Two-phase bubble flows have been used in many technological and energy processes as processing oil, chemical and nuclear reactors. This explains large interest to experimental and numerical studies of such flows last several decades. Exploiting of optical diagnostics for analysis of the bubble flows allows researchers obtaining of instantaneous velocity fields and gaseous phase distribution with the high spatial resolution non-intrusively. Behavior of light rays exhibits an intricate manner when they cross interphase boundaries of gaseous bubbles hence the identification of the bubbles images is a complicated problem. This work presents a method of bubbles images identification based on a modern technology of deep learning called convolutional neural networks (CNN). Neural networks are able to determine overlapping, blurred, and non-spherical bubble images. They can increase accuracy of the bubble image recognition, reduce the number of outliers, lower data processing time, and significantly decrease the number of settings for the identification in comparison with standard recognition methods developed before. In addition, usage of GPUs speeds up the learning process of CNN owning to the modern adaptive subgradient optimization techniques.

1. Introduction

Developing of such an important industrial sector as oil and gas, aerospace, energy inevitably linked with the usage of modern equipment. Unsteady two-phase bubble flows can be found in industrial apparatus and technical devices applied in these areas. In order to successfully tackle with technical challenges in optimizing of processes in existing setups and creating of the new equipment fundamental knowledge is needed on which such solutions can be based. It is necessary to develop precise and effective experimental methods for research of the complex processes.

During the analysis of experimental data in a two-phase flow, the separation of the obtained information on each phase is usually carried out. This helps to analyze a phase behavior independently from each other as they show different properties and study their mutual influence on the mixing efficiency and the turbulence level inside a flow [1].

Among the used non-intrusive methods for studying two-phase flows we can identify the two most promising shadow photography and planar laser techniques, for example as Planar Fluorescence for Bubbles Imaging (PFBI) [2]. The shadow technique has less spatial resolution along the depth, because a measurement area is recorded throughout the full depth and the localization depth is determined by optics' depth of field [3]. The second technique PFBI has a better localization depth and the spatial resolution is defined by a laser sheet thickness used for the illumination of the measurement plane. This gives an opportunity to increase the available range of the measured volume

void fraction inside a flow up to 3% compared to the shadow technique where the upper limit can be assessed in 1-2%.

In order to obtain such parameters of a bubble flow as a velocity field and a distribution of a void fraction, we need numerical methods that are able to extract information on the spatial distribution of bubbles within the flow and their size distribution from experimental data.

The first approach is based on bubble search by edge identification on images and further recognition of the unilaterally connected area for the calculation of a bubble center. The second method is based on correlation search of the bubbles using a bubble template. For both methods, we can highlight common shortcomings: a large number of adjustable parameters (close to 20) and the specialization of algorithms to a particular kind of experimental data. Additionally, we can point out inability of the algorithm based on the edge identification to deal with the overlapped bubbles and the high computational cost inherited to the correlation analysis, though it is able to identify overlapped bubbles as separate objects.

Based on the above arguments, the aim of this study can be formulated as to create a fast and easy to use processing tool of data that was obtained during an experiment by optical diagnostics of two-phase bubble flows. The tool should have the ability to adapt itself to the variation of parameters of the registered bubble images and to deal with the overlapped bubble images.

For the first time to achieve the aims stated above we propose to use a method of artificial neural convolution networks in this study. This requires solving the following tasks:

- creation of algorithmic neural network approaches for recognition of the images of the disperse phase inside bubble flows, which will be easy to adjust to different experiment modifications and mainly they will be able to process overlapped bubble images;
- simulation of the realistic bubble images mimicking experimental data for creation of synthetic images that are needed for training of neural networks;
- implementation of the algorithms for processing of experimental images obtained by PFBI method of quantitative laser visualization.

Today many technical achievements such as speech recognition and face recognition in photographs, well-known machine translator, robots that can redraw paintings - all of this and more is made possible through a deep learning [4]. This is one of the directions of machine learning of an artificial intelligence, which has spread in the last decade thanks to the rapid development of computers and technology.

Briefly, the concept of deep learning can be described as a set of algorithms that attempt to simulate low and high-level abstractions in data by using architectures composed of multiple successive nonlinear transformations.

To extract features from images there have been developed a huge variety of methods, such as SIFT descriptors [5] or histogram of oriented gradients (HoG) [6]. However, all these methods without exception have a large number of settings that greatly complicated the work of a researcher. The concept of deep learning proposes to replace all these complicated and long procedures with learning algorithms. Thus, the basic principle of the method of deep learning can be formulated as the use of neural network architectures that are "trained" how to extract information from data.

The section 2 of this paper provides a brief theoretical presentation of the neural network approach and the methods used. In subsection 2.2, we describe architecture used in this study of the of the so-called convolutional neural networks that are the most advanced technologies for deep learning these

days. Further in the next section 3 we present a description of an experimental setup and parameters for data acquisition. The last section 4 refers to the obtained results and their analysis.

2. Neural networks

2.1. Simple neural network

Figure 1.a shows a simplest implementation of a modern perceptron [7], where the signals $(y_1, \dots y_k)$ of each output neurons are generated by consistent transmission of the input signals $(x_1, \dots, x_d)$ from the neurons in the input layer the l neurons of the hidden layer towards the output layer. The artificial neuron itself has structure [8], shown in the figure 1.b. Each j-th neuron calculates the weighted sum of its inputs v_j with considering the bias b_j through activation function φ. Thus, the signal obtained by neuron goes to his single output. Artificial neurons are combined in a network so, that the outputs of some neurons are connected to the inputs of others, as it shown in figure 1.a.

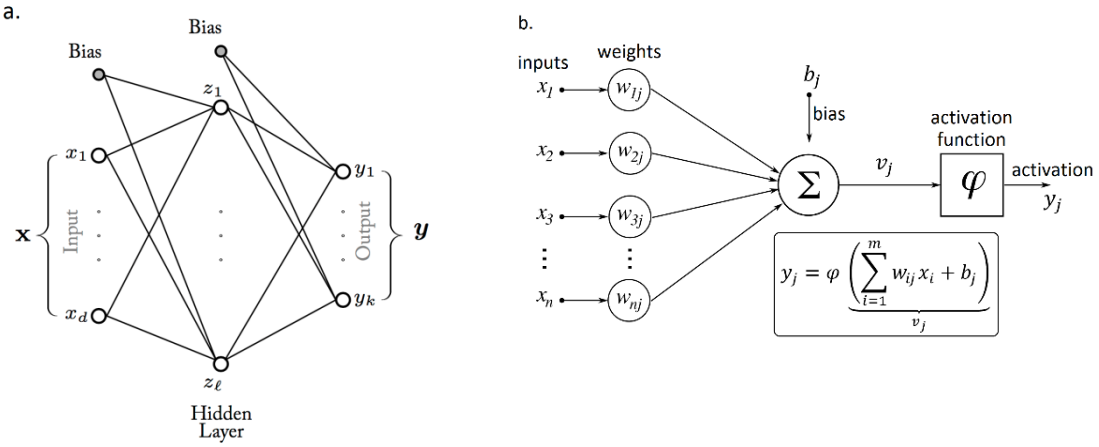

Figure 1. (a.) Single-layer perceptron, and (b.) the structure of an artificial neuron.

2.2. Convolution neural networks

One of the major deep-learning [4] technology, the *convolutional neural networks* (CNN), is aimed at processing graphic information: facial recognition on photos, handwritten digits, etc. Namely because of this basis, to address the problem of identifying bubbles in the images the CNN technology was used in this work. The idea of the convolutional networks was developed together with a group of neuroscientists and the programmer Yann LeCun. The work is based on the study of the principles of operation of the human's visual cortex [9-12]. Like the brain, the convolutional network passes information through several hierarchical layers of pre-processing where each layer performs a specific task. There are three main layers that constitute the CNN.

The convolution layer (figure 2.a) is constructed in such a way that graphically encodes some feature, such as the presence of a sloping line at a certain angle. Then the next layer, formed because of convolution operations with a weights matrix, shows the presence of the sloping line in the treating layer and its position, forming so-called feature maps. The convolutional neural network does not have one set of such weights but a whole range of them, encoding all kinds of lines and arcs from different angles. However, such convolution kernels are not given in advance rather than formed independently through training the network by the classic method of backpropagation [13,14,15,11].

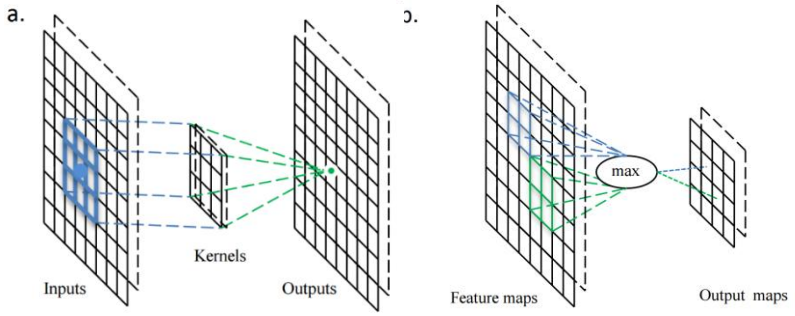

Figure 2. (a) Illustration of the operation principle of the convolution kernel convolutional layer and (b) the subsampling layer. On the left on each of the figures, the feature maps of $n-1$st layer are displayed and on the right the feature maps of n-th layer are depicted, obtained after the using of the corresponding operation.

Subsampling operation layer (figure 2.b) performs a reduction of a dimension of the formed feature maps. The subsampling refers to the process when from the several neighboring neurons of a feature map one is selected that has the maximum output value (max-pooling). Later, it forms one neuron of the next feature map with the reduced dimensions. Due to this operation, in addition to accelerating the further calculations, the network becomes more invariant to the scale of the input image.

Ordinary neuron layer – it is the fully-connected feedforward hidden layers that were previously described in subsection 2.1.

Thus, such architecture of CNNs is interleaving of convolutional, subsampling and ordinary layers of neurons, often namely in that order (see figure 3). Also, sometimes instead of a single layer, several consecutive layers of convolution or sub-sampling are used. Next, neural networks will be designated according to their main hyperparameters. For example, the convolutional network architecture in figure 3 will be marked as *CNN_8_16_50*, where the values 8 and 16 denote the numbers of the feature maps in the first and second convolutional layer respectively, and the number 50 corresponds to the size of the penultimate layer of fully-connected neurons.

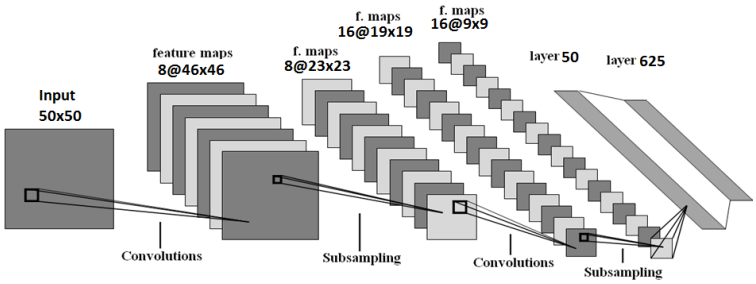

Figure 3. An example of the architecture of the convolutional neural network used in this work.

3. Experiment and used processing image techniques

3.1. Experiment setup

In order to train and test the developed artificial neural network on real experimental data, we visualized a bubbly free jet by PFBI technique (for details refer to [16, 17]). The bubbly jet flow was organized in a hydrodynamic rig. The experimental setup is of closed type and continuous operation. Its test section is made of organic glass with the following dimensions (HxWxL): 425x300x300 mm. The operating fluid was distilled water, which was driven by a centrifugal pump equipped with a control module to vary the rotor speed. The flow rate was measured by an ultrasonic flowmeter. The

water temperature was maintained constant at 30±0.1 °C in the test section by means of a thermostatic regulator filled with tap water as a heat-transfer agent and containing a tubular electrical heater, cooling circuit and aquarium pump to permanently blend the heat carrier. Cooling and heating in the thermostat was implemented by a PID-control system activating an electromagnetic valve to start/stop the coolant supply to the cooling circuit and a switching relay to loop/break the electric line to the heater. The liquid temperature was measured by heat-variable resistors placed in the test section, heat exchanger and settling tank.

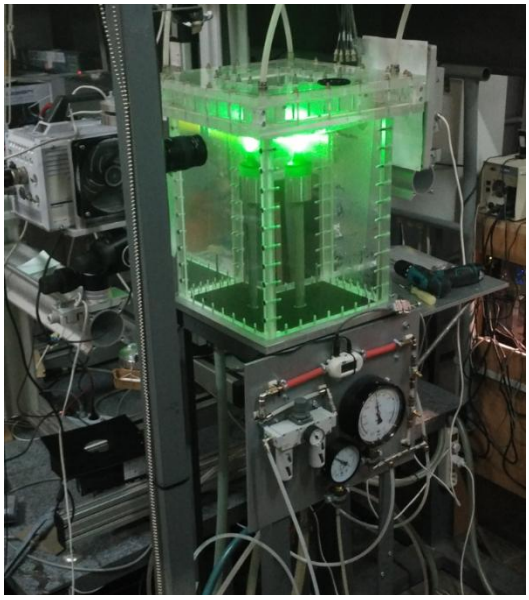

Figure 4. An example of the architecture of the convolutional neural network used in this work.

The submerged free jet was formed by a round contraction nozzle of 110 mm height with an inner diameter at its outlet edge D_N = 15 mm (see figure 4). The ratio of the inlet and outlet cross-section areas of the nozzle was 16. The Reynolds number based on the mean flow velocity $U_0 = 4\,Q_W/(\pi \cdot D_N^2) = 0.663$ m/s, where Q_W is the volume flow rate of the water (measurement error is 2%), and D_N equaled to 12,500. In order to saturate the flow with bubbles, an air-water mixer of a special design was used to provide a quasi-monodisperse size distribution of the bubbles (e.g., see figure 9.a). The air was supplied to the mixer by a compressor through a couple of filters of rough (5 μm) and fine (0.01 μm) cleaning. The air overpressure before the mixer was specified in the range of 9.8–19.6 kPa depending on the air volume flow rate Q_A by a high-precision reducer. The air temperature, pressure and flow rate were measured by a thermal mass flowmeter (TSI model 4140) with the measurement uncertainties 0.1 °C, 0.1 kPa and 2%, respectively. The length of the pipe with 20 mm inner diameter between the nozzle and mixer was 0.56 m. Different volume gas fractions $\beta = Q_A/(Q_W + Q_A)$ in the jet were achieved by changing Q_A using a needle valve. In experiments, β possessed four values: 0, 1, 2 and 3%. The mean diameter of the bubbles D_B was approximately 0.75 mm for all β.

For planar fluorescence imaging (PFBI approach), the water was merged with Rhodamine 6G as a fluorescent dye, the concentration of which was very low (about 20 μg/l). So, the water properties, especially viscosity and surface tension, can be considered unchanged. In order to illuminate and register the bubbles were suspended in the flow. PIV-system with a high temporal resolution

consisting of a pulsed Nd:YAG Photonics Industries DM-532-50 laser (wavelength 532 nm, pulse, pulse energy 5 mJ and pulse duration 160 ns at repetition rate 10 kHz), Photron FASTCAM SA5 CMOS-camera (digit capacity 12 bits, resolution 1,024x1,024 px, acquisition rate 7 kHz) equipped with a Nikon AF Micro-Nikkor 60 mm f/2.8D lens and a low-pass optical filter (transmission edge at 570 nm) and Berkeley Nucleonics Corporation pulse/delay generator (model 575) for external synchronization was used. The thickness of a laser light sheet was about 0.8 mm in the measurement region. The distance between the camera and the measurement plane was roughly 360 mm. In the experiment, the recording rate was 2 kHz. The raw data (a series of 10,000 PFBI snapshots for each regime) was obtained continuously during 5 seconds.

3.2. Algorithm for training images generation

As it was described in subsection 2.2, in order to train a neural network, it is necessary to have training samples. In this work we propose to use the synthetic images, simulating the experimental data, for training the CNNs to search bubbles in images and determine their geometric centers. It is possible to generate the required number of test samples automatically with the known correct responses in advance. For training, the appearance of a bubble image in a square interrogation area of an image was simulated (see figure 5). A situation of hitting of the center of a bubble within an interrogation area was modeled in order that the algorithm could give an accurate assessment of the position of the bubble center. In case of the image of the bubble center went outside of the area, so that part of the bubble image was within the area or totally out of it, the algorithm should give a negative answer. The current situation with the overlapped bubble images was not modeled in this work. The image of a bubble was drawn as an ellipse with the specified center, values of the semi-axes and the rotation angle of semi-major axis with respect to the horizontal direction. The rim of a bubble was added to a black background and superimposed with Gaussian noise and images of tracer particles in the form of uniformly distributed impulse noise, smoothed by Gaussian filter with the kernel size equaled to 1. The intensity distribution profile of the bubble rim by its radius was generated based on the physical model described in [2].

Then a normalized brightness map of the generated image is used as the input data. As for the output data, we used a value map of the following function:

$$P(x,y) = e^{-\left(a\ x-x_0\right)^2-2b(x-x_0)(y-y_0)+c(y-y_0)^2)} \cdot \{\mathbf{1}\ if\ image\ has\ bubble, \mathbf{0}\ -\ otherwise\} \qquad \mathbf{1})$$

- the function of the normal Gaussian distribution (normalized to unity), which reflects the probability of finding a pixel close to the geometric center of the bubble x_0, y_0). All of the coordinates in 1) are measured in pixels px) of a Cartesian coordinate system with the origin at the lower left corner. The coefficients in (1) are responsible for the rotation of the distribution by an angle θ (the angle corresponding to the rotation of the ellipse on the generated image), and are calculated as follows:

$$a = \frac{\cos(\theta)^2}{2\sigma_x^2} + \frac{\sin(\theta)^2}{2\sigma_y^2}; b = \frac{\sin(2\theta)}{4}\left(\frac{1}{\sigma_y^2} - \frac{1}{\sigma_x^2}\right); c = \frac{\cos(\theta)^2}{2\sigma_y^2} + \frac{\sin(\theta)^2}{2\sigma_x^2} \quad \mathbf{2})$$

The values of the dispersion parameters σ_x, σ_y in 2) were chosen so that the function 1) satisfyied the condition:

$$P(\tilde{x},\tilde{y}) \le 0.01\ \forall\ \tilde{x},\tilde{y}) \in\ the\ rim\ of\ the\ drawn\ bubble\ (\mathbf{3})$$

Figure 5. Examples of the generated synthetic images for training of the neural network.

3.3. Bubble detection algorithm

The processing algorithm of experimental images operated as described below in the algorithm 1. A square window moved through the experimental image, scanning by the trained neural network. In each window, if a bubble was found, then the most probable coordinates of its center were marked. Then after a pass through the entire image, obtained nearby centers were united, according to their probability weights, as belonging to the single physical bubble. Moreover, centers of a bubble $\mathbf{x_1} = x_{11}, y_{12})$ и $\mathbf{x_2} = (x_{21}, y_{22})$ were considered close to each other if $\| \mathbf{x_1} - \mathbf{x_2} \|_{L_2} \leq R$, where R is a parameter, depending on gas volume void fraction of the flow. On average, for the gas volume void fraction of $3-4\%$, the optimal value, which was obtained during several computational experiments, equals $R \approx 15\ px$.

The algorithm also uses probability thresholds $tresh_1$, $tresh_2$, characterizing the probability p_0 and p_{xy} to consider that object in the current window is a bubble and consider whether that (x, y)-th pixel is inside the bubble shape, respectively. Generally speaking, the optimum values of these threshold parameters (which was confirmed by the experiment) can be defined numerically on the basis of characteristics of the probability distribution (3) as $\overline{tresh_1} \approx 0.2; \overline{tresh_2} \approx 0.23$ (considering that the average bubble radius is about $15-20$ pixels).

Additionally, the option of using regression models (a nonlinear least squares method and its modifications) in order extract the unknown sizes and the rotation angle of a bubble by fitting the distribution of the output of the neural network was tested. However, this method was not used further because of the necessity to specify starting parameters of an approximating distribution, long calculation time (on the order of seconds for a window of 25x25 units), and, most importantly, due to the high probability the algorithm's divergence.

1. In an input image, a window with the size 50×50 px is moved;
   A parameter ***s*** (in px) of the window shift is fixed;
2. In the w_i-th window the $\mathrm{p_0^i}$ threshold is calculated by the CNN's outputs:
   $$p_o^i = \frac{1}{N} \sum_{x,y)\in W_i}^{50\times 50} p_{xy}^i \cdot \begin{cases} 1, p_{xy}^i \geq \boldsymbol{tresh_2} \\ 0, p_{xy}^i < \boldsymbol{tresh_2} \end{cases}; N = \sum_{x,y)\in W_i}^{50\times 50} \begin{cases} 1, p_{xy}^i \geq tresh_2 \\ 0, p_{xy}^i < tresh_2 \end{cases}$$
3. ***If*** $\mathrm{p_0} > \mathbf{tresh_1}$, ***then*** the $\mathrm{W_i}$ contains a bubble with center $x_0, y_0) = argmax_{\ x,y)\in W_i}(p_{xy}^i)$;
4. Connection of k nearby centers $\{(x_{0j}, y_{0j}), j = 0 \ldots k\}$ into the single one:

$$x_{0i}, y_{0i}) = \frac{1}{k \cdot p_{max}^i} \sum_{j=1}^{k} (x_{0j}, y_{0j}) \cdot p_0^j; p_{max}^i = \max_{j=0\ldots k}(p_0^j);$$

Algorithm 1. The bubble detection algorithm.

3.4. Description of the finally chosen method

In this work for both of the above tasks, about 50 various architectures of the CNNs and the MLPs was tested in each group respectively. Selection of hyperparameters (layers' sizes, learn rate etc.) conducted automatically, using the Bayesian optimization method [18]. The experimental framework is implemented on C/C ++. In order to simplify the program's user interface, the Python wrapper has been written, and all "mathematics" - has been implemented using C ++ linear algebra libraries BLAS/LAPACK to improve performance. To prevent overfitting, it used a variety of methods [19] - it managed to obtain the absolute additive in accuracy in the test on the synthetic images about 7-9%.

The experimental data set consists of 12,000 images of 50×50 px, generated as it was described in subsection 3.2. Input vectors for both networks $\left(x_{11}^d, \ldots, x_{5050}^d\right)$ are a two-dimensional array of brightness values of each pixel: $x_{lm}^d \leftrightarrow I(x_l, y_m)$. The output vectors $\left(t_1^d, \ldots, t_{625}^d\right)$ are an unwrapped one-dimensional array of probabilities: $t_i^d \leftrightarrow P(x_i, y_i)$. We used the following algorithms: *Adam, AdaDelta, AdaGrad, SGD* and their various modifications [15] during the computational experiments of determination the optimal settings for the neural network training as the optimizers of the functional quality. The error function was the log likelihood function, which showed, in average, slightly better convergence speed (5-10% faster) of minimization of the quality function [15], than the quadratic error function.

Table 1. Examples of the studied neural networks.

Architecture	Parameters ($\times 10^3$)	Weight (KB)
MLP_100_100_100[a]	335	1298
MLP_256	800	3100
CNN_8_16_50	100	380
CNN_8_16_100	196	754
CNN_10_20_100	230	880

[a] Multi-layer perceptron (or *DNN*-deep neural network) with 100 neurons in hidden layers.

4. Results

4.1. Processing of synthetic images

Hereinafter, the results of a pair of the best tested architectures are presented for each of the groups, respectively. The next two figures of neural networks tests were obtained on the test data set of 1000 synthetic images through the training with the following experimental parameters:

- CNN: SGD-Nesterov with momentum 0.9
- DNN: SGD with momentum 0.9
- Examples/batch = 10; learn rate = 0.0003; 1000 test samples and 11000 train samples
- *GPU*: Nvidia GeForce GTX 980 Titan Z, 6 Gb RAM
- *CPU*: Intel Core i5 - 2.7 GHz; SDD

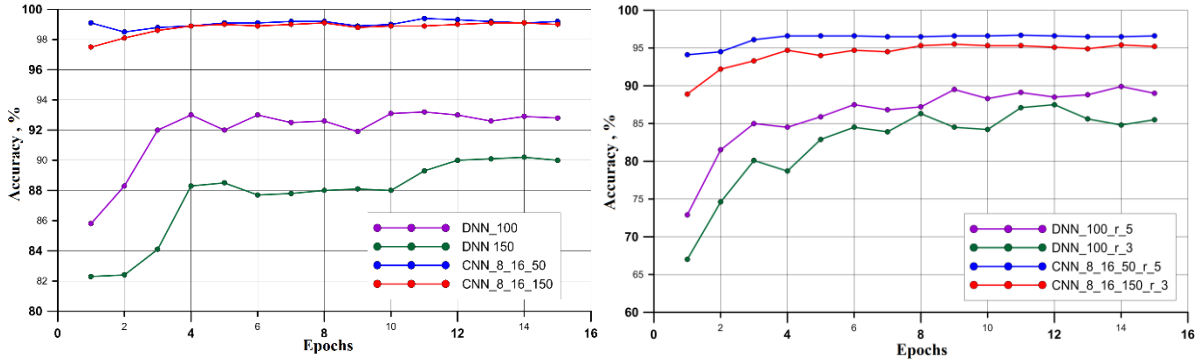

Figure 6. Correctly recognized bubbles on synthetic images (left) and correctly identified bubbles' geometrical centers with three pixel accuracy (right).

It is seen from figure 6 that the convolutional networks provide much better results than DNN networks. They show themselves better also in the problem of determining the exact center of the bubble. The accuracy of determining the size and orientation of the bubble (average for three parameters a, b and θ value in %) for different architectures almost completely (up $\approx 0.5\%$) coincides with the corresponding lines in figure 6 right. The following graph (figure 7) reflects the rate of convergence of different optimization methods for error of log-likelihood function for the first problem.

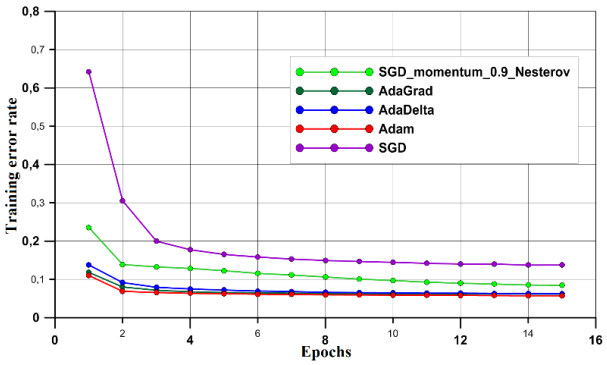

Figure 7. Minimization of the quality function by the various optimizers during the training of the CNN_8_16_50.

Algorithm	Epoch number[a]	Accuracy, %
Adam	**1**	**94.1**
AdaDelta	3	91.8
AdaGrad	2	90.3
SGD-Nesterov	7	93.2
SGD	11	91.6

Table 2. Efficiency of the optimization algorithms for CNN_8_16_50[a]. Epoch number, when the accuracy on the test data exceed 90%.

The Adam algorithm showed itself as the most effective (tables 2 and 3) during testing. The table 3 shows the training time in seconds of the network on CPU and GPU. It is clear from the graphs that on GPU the convolutional network learns very quickly, which allows the researcher to generate new types of training images and almost immediately verify the effectiveness of this new model.

Thus, after the numerous experiments the architecture CNN_8_6_50 (figure 3) was identified as the most effective. It is also worth to note, that on average the usage of GPU with parallelized calculations speeds up the training process up to 8-10 times. Subsequent analysis of the experimental data was carried out precisely by this neural network. Table 3 shows the best obtained the results for the trained CNN_8_16_50 model during synthetic image processing in the course of the experiments.

Table 3. Time of training the CNN_8_16_50 for various optimizers, seconds

Algorithm	CPU[a] (15 epochs)	CPU[b] (accuracy>90%)	GPU[a] (15 epochs)	GPU[b] (accuracy>90 %)
Adam	755	**48**	127	**9**
AdaDelta	740	100	124	30
AdaGrad	735	95	123	17
SGD-Nesterov	710	322	117	53
SGD	**690**	506	**114**	89

[a] Total time spent for the appropriate number of iterations;

[b] Total time necessary for reaching the respective accuracy.

Table 4. Final results on the synthetic images.

Identification accuracy of a bubbles, %	Center of bubble definition accuracy, %
99.8	**96.4**[a]

[a] With precision of 3 pixels;

4.2. Comparison of the processing performance

Table 5 shows average values of the total time required for processing of an image with the size of 1280x1024 pixels for the CNN_8_16_50 neural network and the correlation algorithm processing time. According to the tests, it was found that on average the proposed method realized in GPU is faster than correlation approach almost for 8-10 times.

Table 5. Processing time of the image with the size 1280 × 1024 px.

Processor	CNN[a] total time, sec	CNN[a] time for window, ms	The correlation algorithm total time, sec
CPU	**54.7**	**5**	≈80
GPU	**8.1**	**0.8**	not parallelized

[a] Window shift size = 10 pixels.

Table 6. Comparison of the correlation algorithm and the network CNN_8_16_50, trained with the accuracy 99.8% on 1000 test images.

Shift, px	$tresh_1$	$tresh_2$	Total number of bubbles	Number of errors[a]	Correctly recognized bubbles[b]
5	0.2	0.15	≈100	**5**	91-93
7	0.19	0.22		15	**94-96**
10	0.12	0.04		20	89-91
15	0.1	0.09		10	88-90
The correlation algorithm				30	80-85

[a] ≈number of unmarked bubbles and marked as bubbles blank areas: *false negative + false positive rates*

[b] ≈number of marked bubbles relative to the total number of bubbles in percentages: *true negative rate*

Table 6 shows the performance of the correlation method and the convolutional network during the processing of the experimental image with the size 1280 × 1024 px (figure 8). Obviously, it is impossible to automatically assess this kind of results and table 6 gives approximate figures for several dozen different experimental images processed manually. According to the results, the neural network identifies more bubbles on average of 10-15% and makes less mistakes in 8-10% of cases. Moreover, accuracy of detection of a center of the bubbles for both methods is almost the same.

4.3. Processing of experimental images

Figure 8 shows the examples of how the same processed image section looks with the different shift values (subsection 3.3 algorithm 1). Only windows with bubbles chosen by the network are shown. The results obtained by the bubble detection algorithm are shown below.

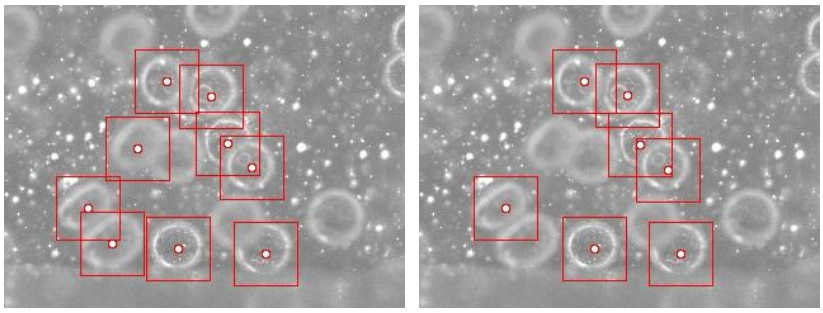

Figure 8. The results of the bubble detection algorithm. The marked centers refer to the interrogation area and not to the bubbles.

Left: $s = 5px$; $tresh_1 = 0.2$; $tresh_2 = 0.03$; *right:* $s = 10px$; $tresh_1 = 0.24$; $tresh_2 = 0.03$.

Figure 9 shows the examples of processing results of the experimental images by the convolution network and the correlation algorithm.

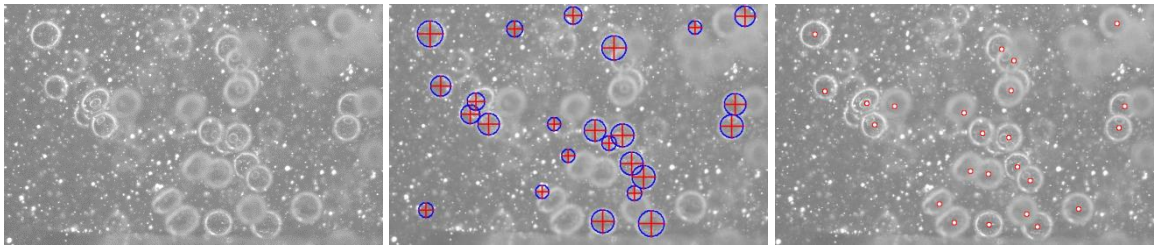

Figure 9. An example of the processing results of a part of an experimental image. *(left):* The original image and *(center):* The processing by the correlation algorithm. *(right):* The processing by the neural network

5. Conclusions

This paper demonstrates that modern technology of artificial neural networks can be successfully applied for experimental data processing of fluid dynamic experiments. As a result of this work, the software was developed and tested based on the deep learning algorithm for bubble center detection and synthetic bubble image generation for a neural network training. A number of different neural network architectures were studied. The CNN was chosen as the most efficient architecture. Preliminary tests of the bubble detection upon experimental images were conducted with the software.

A key aspect of our study is the end-to-end system including creating synthetic data set for training of the fully customizable convolution neural network model and implementation of methods for validation the trained models. Our machine learning based approach showed significantly faster processing time and better accuracy compared with existing approaches. Time for processing a standard image of 1280x1024 pixels takes about 8 seconds on GPU unlike 80 seconds to be taken by the CPU-based not parallelized correlation algorithm. Moreover, the neural network identifies 10-15% more bubbles on average and makes less mistakes in 8-10% of cases.

Future work will be focused on addition of the overlapped bubbles for synthetic training set and the improvement of physics of bubble simulation as well as developing reliable methods for detection of bubble sizes.

Acknowledgments
The work was financially supported by the Russian National Foundation (Grant No. 14-19-01685, guided by D.M. Markovich).

References

[1] Lance M, Bataille J (1991) Turbulence in the liquid phase of a uniform bubbly air–water flow. J of Fluid Mech. 222. pp 95–118
[2] V.M. Dulin, K.S. Pervunin, D.M. Markovich. A Technique for Round Bubbles Imaging via Planar Fluorescence // 17th International Symposium on Applications of Laser Techniques to Fluid Mechanics. Lisbon, Portugal, 07-10 July, 2012
[3] Lindken R, Merzkirch W (2002) A novel PIV technique for measurements in multiphase flows and its application to twophase bubbly flows. Exp Fluids 33. pp 814 – 25
[4] I. Goodfellow, Y. Bengio and A. Courville. Deep Learing. // An MIT Press book. 2016
[5] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints.// Computer Science Department. University of British Columbia, Vancouver, B.C., Canada. 2004
[6] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. // INRIA Rhone-Alps, Montbonnot 38334, France. 2004
[7] F. Rosenblatt Principles neurodynamics. Perceptron and the theory of brain mechanisms. // M.: Mir, 1965. p 480
[8] Osovsky C. Neural networks for processing information. Trans. from Polish I.D.Rudinskogo. // M .: Finance and Statistics, 2002. p 344
[9] Y. Le Cun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, et al. Handwritten digit recognition with a back-propagation network. In Advances in neural information processing systems, 1990.
[10] Y. LeCun, F.J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, volume 2, pages II–97. IEEE, 2004.
[11] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision.
[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition.// Proceedings of the IEEE, 1998.
[13] Kingma, D., and Ba, J. “Adam”: A method for stochastic optimization. // In International Conference for Learning Representations, p 201

[14] Nesterov Yu E. Methods of the convex optimization. // MTNMO, Moscow. 2010
[15] O. Dozat. Optimizing CNNs. Stanford // ArXiv preprint. 2016
[16] Akhmetbekov Ye.K., Alekseenko S.V., Dulin V.M., Markovich D.M., Pervunin K.S. Planar fluorescence for round bubble imaging and its application for the study of an axisymmetric two-phase jet // Experiments in Fluids. 2010. V. 48, No. 4. pp 615 – 29
[17] Dulin V.M., Markovich D.M., and Pervunin K.S. The optical principles of PFBI approach AIP Conf. Proc. 1428. 2012. pp 217–24
[18] Z. Brochu, V. M. Cora, N. de Freitas. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. // ArXiv preprint. 2010
[19] N.Srivastava, G. Hinton, A. Krizhevsky et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. // Department of C.S., University of Toronto. Toronto, Ontario, M5S 3G4, Canada