TFC2022

TRIZ and Progressive Artificial Neural Networks

Vladimir Proseanic, Boris Zlotin

In 1958, psychologist Frank Rosenblatt built a working brain model that became the world's first ANN - an artificial neural network called the perceptron. Because some essential biological patterns were unknown at that time, Rosenblatt made two fatal errors that all classical neural networks have shared. These errors caused numerous problems in modern ANNs and made it impossible to create practical Artificial Intelligence.

TRIZ application ensured the detection of fatal ANN errors.
This allowed TRIZ specialists D. Pescianschi, V. Proseanic, and B. Zlotin to develop the concept of a fundamentally new neural network devoid of the most dangerous problems and shortcomings of classical ANNs. These networks, created by Progress Inc. called Progressive Artificial Neural Networks (PANN)

Based on PANN networks, several software products have been developed that have been successfully tested, thus confirming and exceeding theoretical expectations. The new products based on PANN are being designed and pre-marketed by Omega Server Inc.

TFC2022

Vladimir Proseanic, Chief Executive Officer, Chairman of the Board of Directors, and co-founder of Progress Inc, Omega Server Inc.

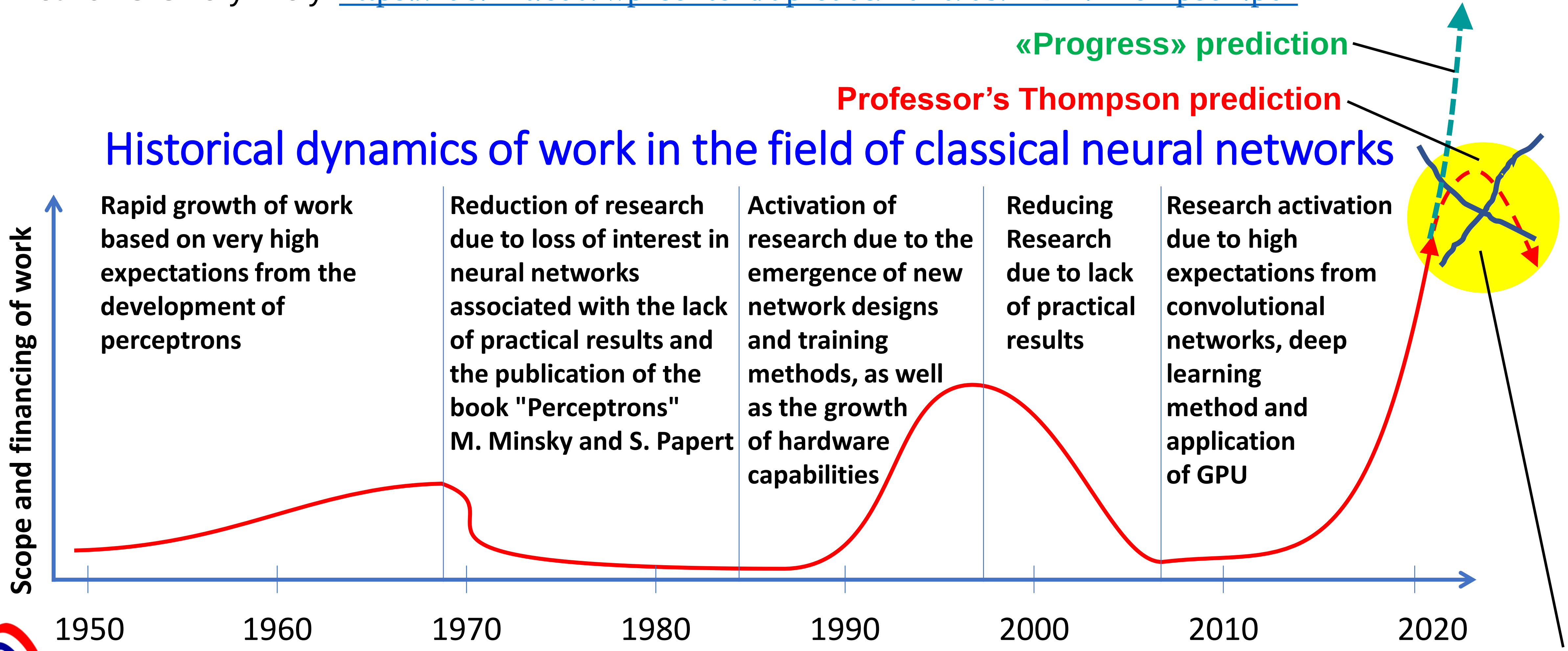
Former Director of Analytical and Intellectual Property Services at Ideation International, Inc., USA, Mr. Proseanic has accumulated diverse leadership and managerial experience as the leader of multiple successful engineering innovation projects and the Director of the world's first private TRIZ company. Under his leadership, Progress Inc. has obtained 14 patents worldwide for the next generation of artificial neural network technology. Omega Server Inc. aims to develop revolutionary digital servers licensed by Progress Inc. for artificial intelligence applications. Trained by the TRIZ founder Genrich Altshuller in the early 1980s, Mr. Proseanic is widely recognized in the TRIZ community as a brilliant inventor, the author of multiple patents, and an active contributor to the science of innovation.

Boris Zlotin, Chief Scientist and co-founder of Progress, Inc

Ms. Zlotin is a certified TRIZ Master with almost 50 years of TRIZ experience, including participating as a fellow instructor in numerous seminars conducted by the TRIZ founder Genrich Altshuller during 1981-1986. He is widely recognized as the leader of the TRIZ community and is considered one of the world's foremost theorists and TRIZ scientists today. He is responsible for most of the advances made to the methodology to date, trained over 10,000 people worldwide, and solved over 30,000 problems. He is a multiple patent holder and author of books, papers, and special publications on TRIZ. Since 2012 has been involved in developing a new type of AI - Progressive Artificial Neural Networks (PANN).

History of the Successes and Failures of Artificial Neural Networks

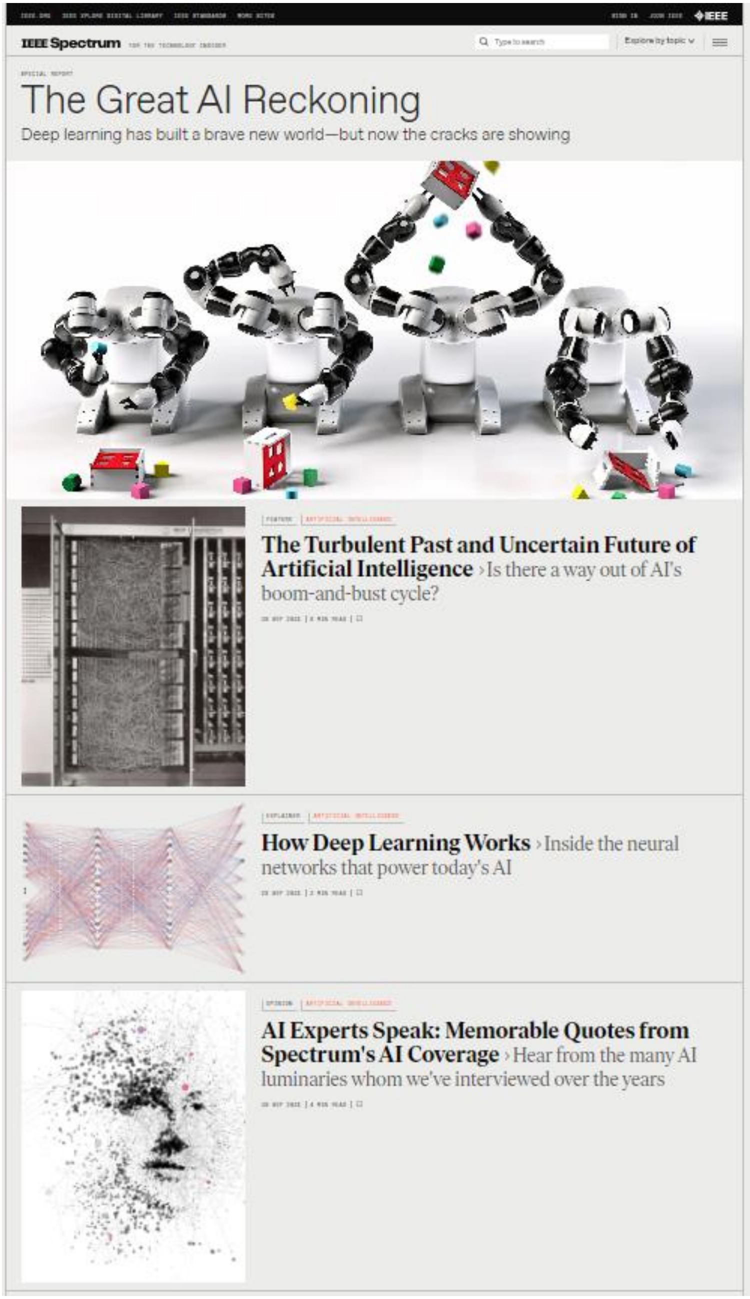
Neil S. Thompson, MIT professor, in the article THE COMPUTATIONAL LIMITS OF DEEP LEARNING, shows that the evolution of AI is on an unstable trajectory, and another severe failure in developing classical neural networks is very likely. https://ide.mit.edu/wp-content/uploads/2020/09/RBN.Thompson.pdf



«IEEE Spectrum» SPECIAL REPORT: ***Why Is AI So Dumb?***

TFC2022

IEEE Spectrum, October 2021
https://spectrum.ieee.org



IEEE Spectrum

SPECIAL REPORT

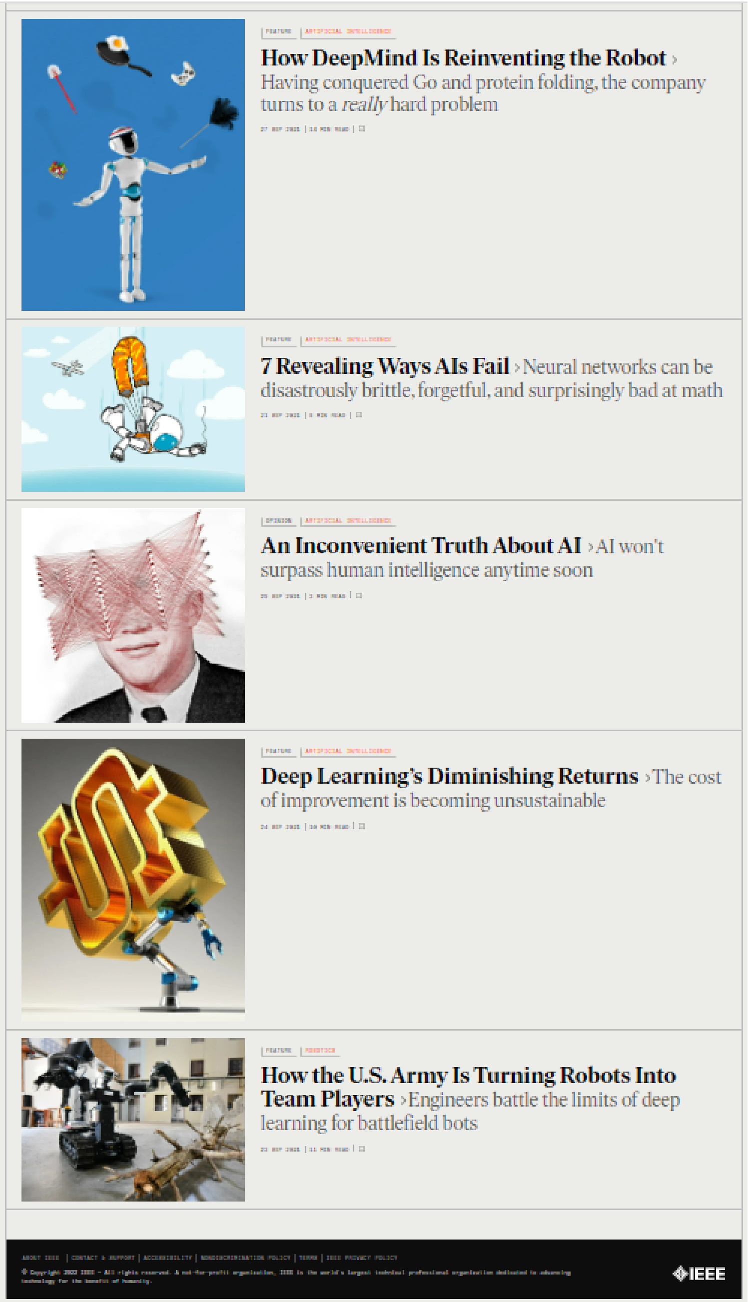
The Great AI Reckoning

Deep learning has built a brave new world—but now the cracks are showing

The Turbulent Past and Uncertain Future of Artificial Intelligence › Is there a way out of AI's boom-and-bust cycle?

How Deep Learning Works › Inside the neural networks that power today's AI

AI Experts Speak: Memorable Quotes from Spectrum's AI Coverage › Hear from the many AI luminaries whom we've interviewed over the years



How DeepMind Is Reinventing the Robot › Having conquered Go and protein folding, the company turns to a *really* hard problem

7 Revealing Ways AIs Fail › Neural networks can be disastrously brittle, forgetful, and surprisingly bad at math

An Inconvenient Truth About AI › AI won't surpass human intelligence anytime soon

Deep Learning's Diminishing Returns › The cost of improvement is becoming unsustainable

How the U.S. Army Is Turning Robots Into Team Players › Engineers battle the limits of deep learning for battlefield bots

IEEE

https://spectrum.ieee.org/special-reports/the-great-ai-reckoning/

Problems of Classical ANNs

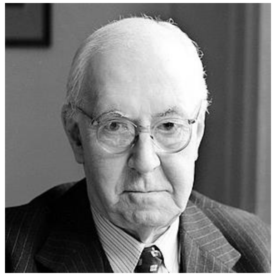
These doctrines created the main issue for all types of ANN - that between all synaptic weights in the training process, strong recursive (mutual) feedbacks arise; a change in each weight affects all other weights. This issue leads to several problems:

- Due to the strong dependence between the weights, a change in one weight generates changes in all other weights, the number of which can reach enormous values. As a result, it is necessary to repeat the entire training cycle for each weight multiple times.
- Training is conducted using methods of "gradient descent," that is, through many small steps of changes in synaptic weights. For complete training, it is necessary to spend thousands, sometimes hundreds of thousands of epochs. That requires enormous computing power and leads to an unacceptable increase in time and cost of developing neural networks and their training time.
- To add at least one new image or eliminate an unnecessary one, complete retraining of the entire network is required.
- In the process of training a classical ANN, its intelligence is formed, that is, the ability to recognize images close to the trained ones. At the same time, with overfitting, it will be effectively recognizing the images that it was trained with but will not be able to recognize images even slightly different from them. https://www.ibm.com/cloud/learn/overfitting; https://www.datarobot.com/wiki/overfitting/

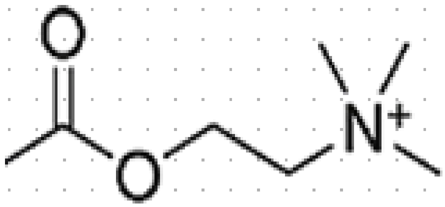
Problems of Classical ANNs

- In the training process, due to recursive feedback, undesirable nonlinear effects may occur, such as the inability to estimate the training time in advance and the lack of a guarantee of successful completion of training due to phenomena such as neural network paralysis, freezing, a local minimum, etc. And the probability of problems increases with the number of neurons, the training volume, and desired accuracy.
- Due to the nonlinear nature of neural networks, enormous difficulties arise in constructing their structure, choosing the optimal parameters, and the practical impossibility of scaling and increasing the power of a trained network.

The cause of ANN problems - Rosenblatt's fatal mistakes



Henry Hallett Dale discovered neurotransmitters in the twenties, for which he received the Nobel Prize in 1936. Based on his work, the "Dale doctrine" was born: **"one synapse - one neurotransmitter."** This doctrine was experimentally refuted in the seventies.



The neurotransmitter acetylcholine

in Rosenblatt's time, Dale's doctrine seemed "absolutely true," and it evolved into what might be called the **"Rosenblatt doctrine": "one synapse uses one and only one synaptic weight."** And this doctrine, which ultimately determined the design of the perceptron and its “heirs,” still reigns in all classical neural networks.

The Dale Doctrine was refuted in the 1970s. Today it is known that in each biological synapse, several transmitters can work, depending on the characteristics of the signal arriving at the synapse. As applied to neural networks, this can be interpreted as the presence of several different weights on one synapse; the choice of which to use should be related to the nature of the input signals. Below we will consider the design of an artificial neuron and the construction of a new type of network based on this principle.

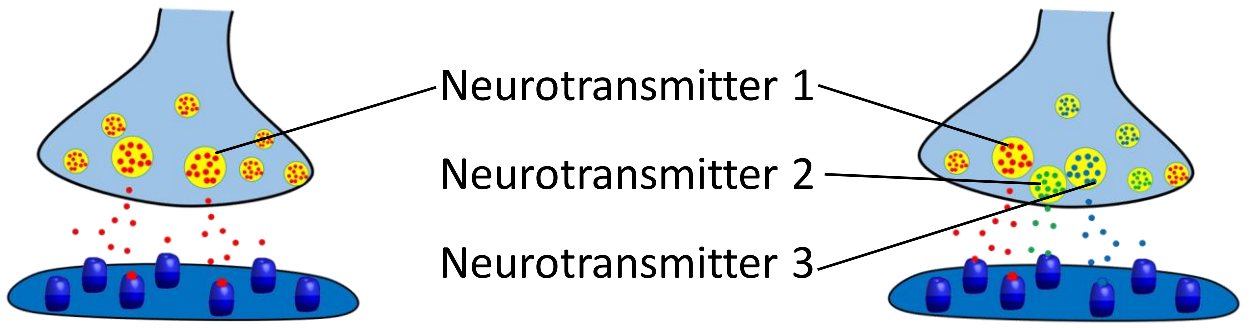
The main differences between artificial neurons of classical neural networks and PANN

1. More than one corrective weight on a synapse
2. Distributor, selecting corrective weight according to the value of input signal
3. No need in activation device

From biological to artificial synapse

Synapse of a biological neuron with one neurotransmitter | **Synapse of a biological neuron with multiple neurotransmitters**

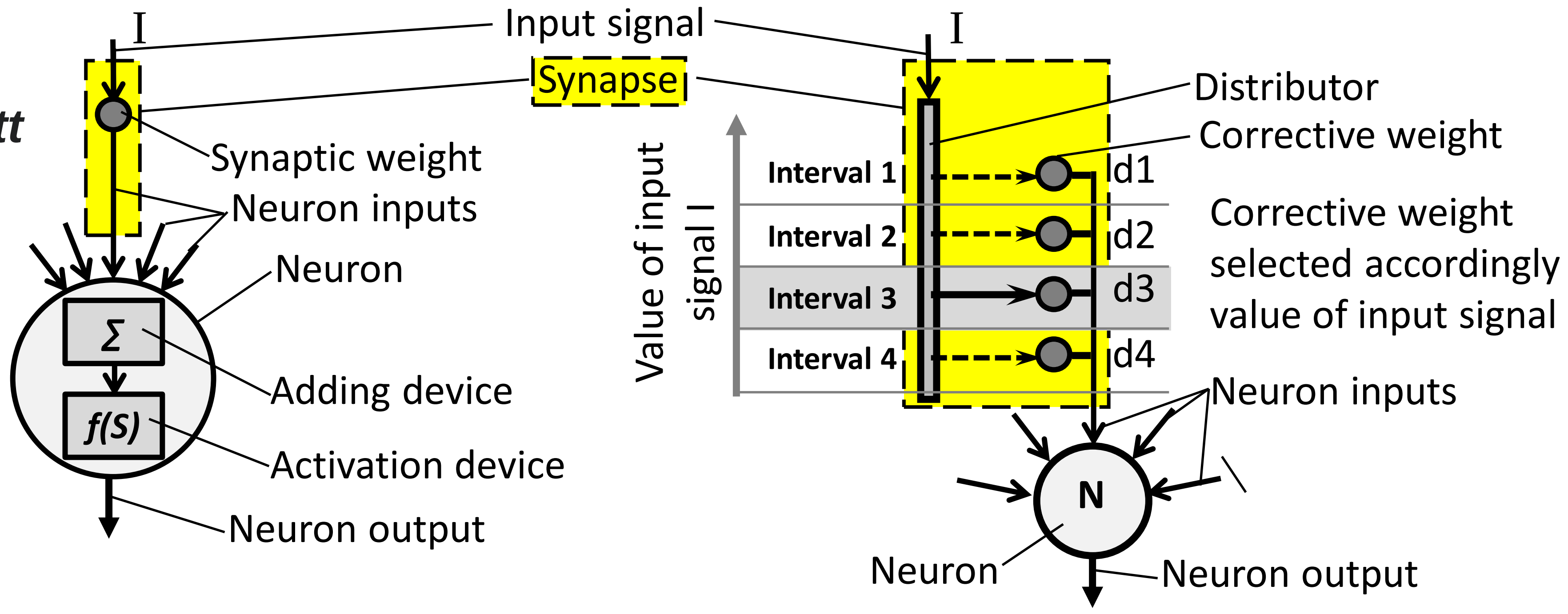
"Dale doctrine"



~~***"Dale doctrine"***~~

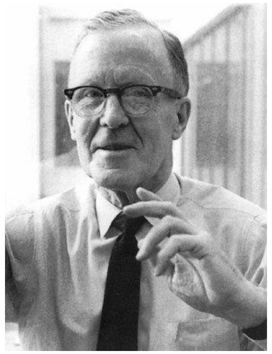
Artificial neuron of classical ANN | **Artificial neuron of PANN**

"Rosenblatt doctrine"

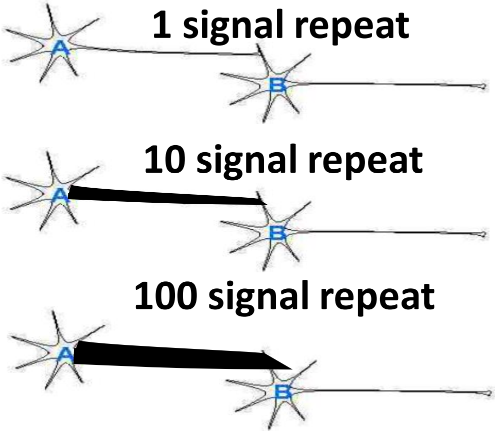


~~***"Rosenblatt doctrine"***~~

The cause of ANN problems - Rosenblatt's fatal mistakes



In 1949 Donald Hebb showed that training a biological neuron is possible by multiple repetitions of signals. This is known as **"Hebb's rule.”** Even in 1949, this doctrine was highly dubious, contradicting the well-known facts of the almost instantaneous training of animals and humans at the first perception of specific signals (for example, those associated with danger, pain, or strong emotions).



Strengthening neuron connections with signal repetitions

Unfortunately, the Hebb doctrine formed the basis of the **gradient descent method created by Rosenblatt for training the perceptron**. Its essence is the repeated supply of training signals to the neural network and the gradual approximation of the network to the state of "learning" in thousands of tiny steps, each of which requires vast calculations. And it is still used with slight modifications in all classical neural networks.

Training classical neural network versus PANN network

Hebb training doctrine

Each synapse has only one synaptic weight, and the network is trained by changing this weight.

There is a vital feedback between all synaptic weights, which causes:

- «Collective training" - taking into account at each step of training the mutual influence of all weights on each other.
- Training the system in successive small steps (gradient descent method)

This requires the solution of large systems of equations, which leads to a long duration of each training epoch and a large number (thousands and even hundred thousand) of epochs.

The training time is roughly proportional to the product of the exponents from the number of neurons (n) and the number of images (m) for training.

$$T \equiv e^n \times e^m$$

PANN training doctrine

Each synapse has a set of two or more corrective weights, and training is done by selecting the “appropriate weight” and changing it.

In the simplest case, there is no feedback between the corrective weights, which allows:

- "Individualized training" - without considering the influence of weights on each other.
- Training the system "in one step” for the entire error value without the gradient descent method

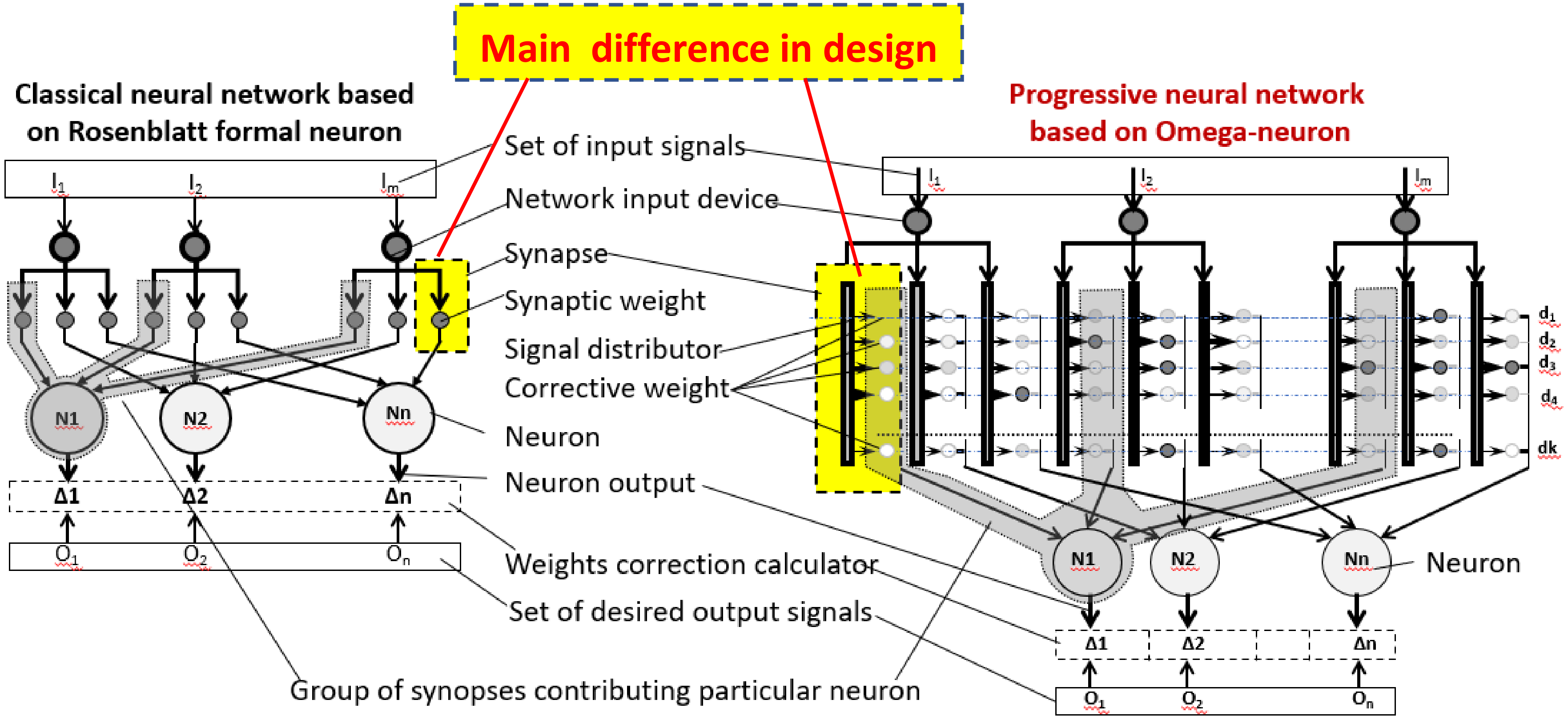
This requires small arithmetic calculations, which leads to short training time.

The training time is roughly proportional to the product of the squares of the number of neurons (n) and the number of images (m) used for training.

$$T \equiv n^2 \times m^2$$

Classical Artificial Neural Network VERSUS Progressive Artificial Neural Network

"Rosenblatt doctrine"



«PANN doctrine"

PANN is mainly similar to the classical network in its main features. There are three main differences:

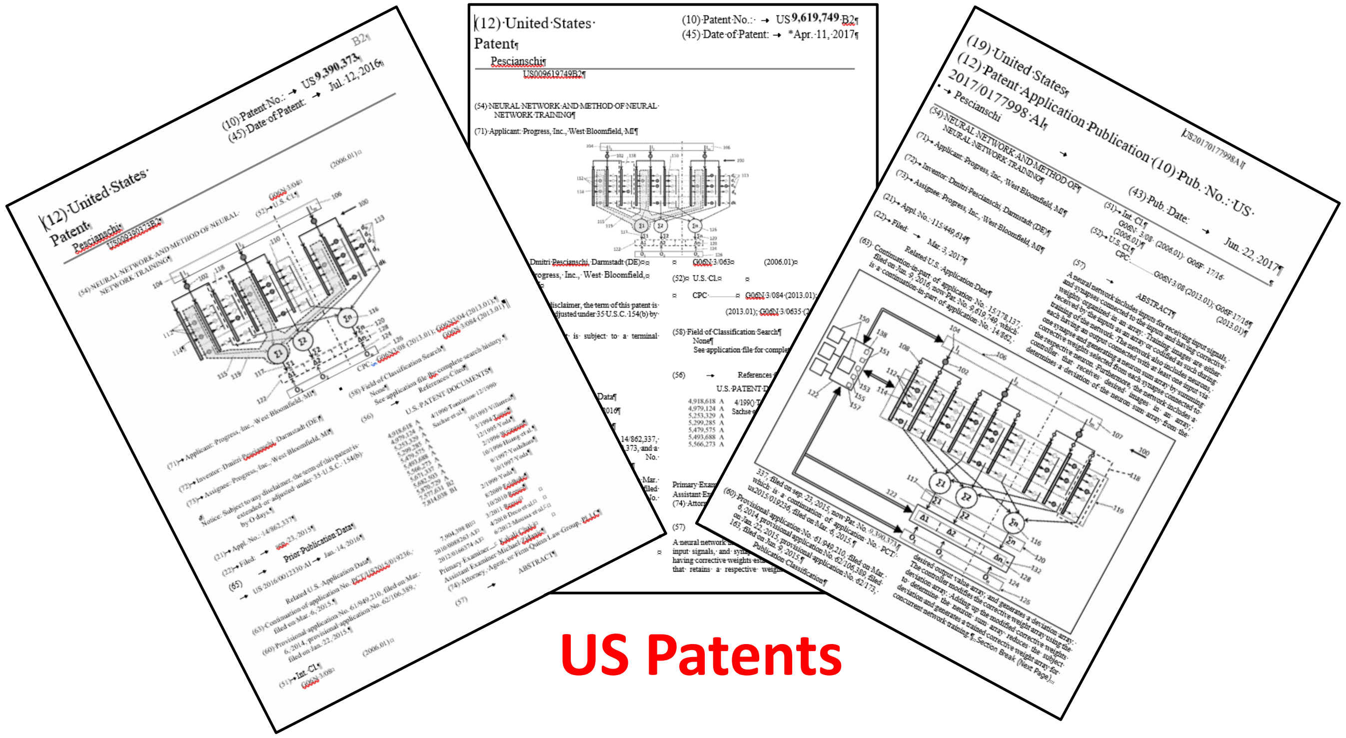
1. Fixed "Dale doctrine" - each synapse has not one but two or more synaptic weights. This weakens the spurious feedback between the weights that cause a massive increase in network training time.
2. Fixed “Hebb doctrine” – network is trained without gradient descent method, which makes a large number of training epochs unnecessary
3. The design of artificial neurons is drastically simplified by eliminating the non-linear activation function, which is superfluous and harmful to PANN. The Omega neuron used in PANN is a simple signal-summing device.

Overcoming Mistakes

The idea of a new neural network with more than one synaptic weight per synapse (contrary to the Dale doctrine) and training without the gradient descent method was proposed by our TRIZ student Dimitry Peschansky. Usually, the path from a breakthrough idea to an actual product is very long. Still, it was possible to pass it concisely thanks to the consistent, purposeful TRIZ application, in particular, the Directed Evolution process and the findings of the Chisinau School of TRIZ in solving research problems. The result of this work was a series of networks called Progressive Artificial Neural Networks (PANN)

Intellectual Property Protection

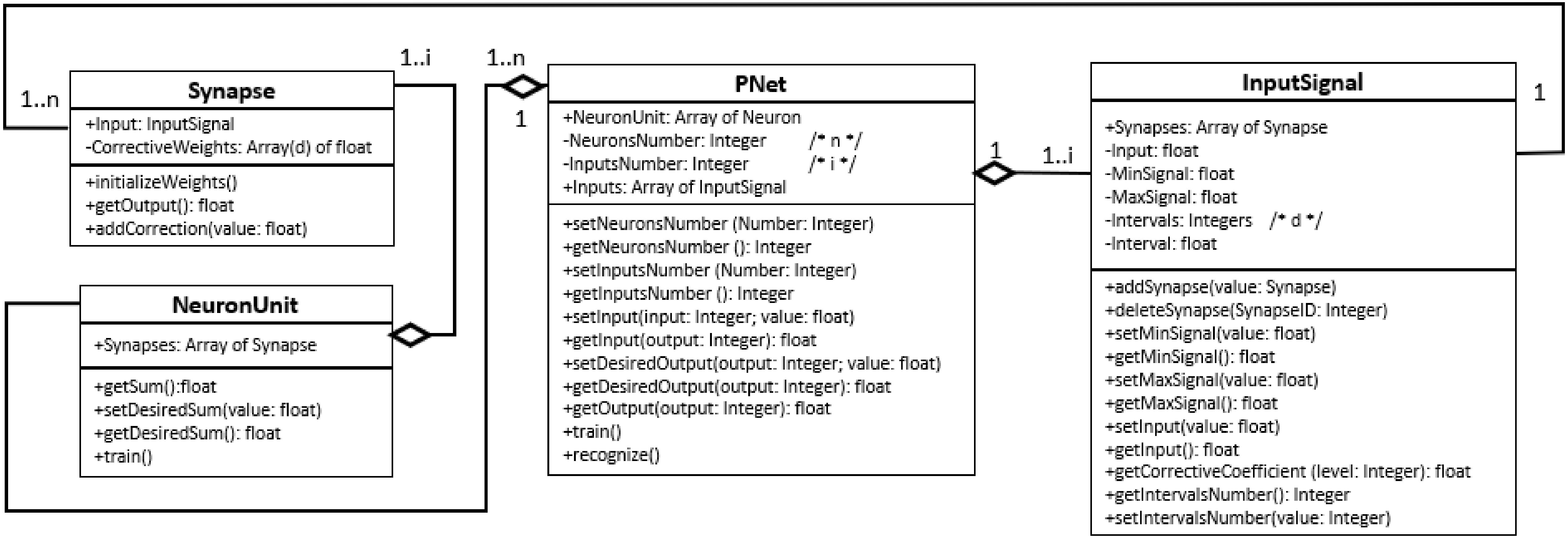
1. **Patent US** 9390373; 2016
2. **Patent US** 9619749; 2017
3. **Patent US** 10423694; 2019
4. **Notice of Allowance** 16/523.584; 2022
5. **Japan Patent** 6382354;2018
6. **China Patent** ZL201580012022.2; 2018
7. **Mexico Patent** MX357374B; 2018
8. **Taiwan Patent** I655587; 2019
9. **Israel patent** 247533; 2019
10. **Hong Kong Patent** HK1227499; 2019
11. **Singapore Patent** 11201608265X; 2019
12. **Eurasia Patent** 035114; 2020
13. **Korean Patent** 10-2166105, 2020
14. **European Patent** EP17811082. 1, 2021



US Patents

TFC2022

UML model for object-oriented programming of PANN



PANN training is carried out in the same way as conventional neural networks, “pixel by pixel.”

From the patent: US 9.390.373

NEURAL NETWORK AND METHOD OF NEURAL NETWORK TRAINING

PANN Batch Training

But the simple theoretical model underlying PANN allows instead **“pixel by pixel”** training to use a more efficient approach - batch, one-shot training. It uses the matrix algebra tools available in all major computer languages.

Step 1. Preparation of training - the formation of data matrices for training. In particular:

- Input image matrix **| I |** - one-dimensional vector including all image pixels
- Matrix of desired output image **| O |** - one-dimensional vector including the desired output signals corresponding to all images selected for training
- Matrix of corrective weights **| w |** - a two-dimensional matrix in which all weights are equal to zero at the beginning of training.
- Matrix of influence coefficients of corrective weights **| c |** - in fact, this matrix acts as a distributor, determining which of the weights on the synapse corresponds to each input image pixel.

Matrix |I|

	Pixel 1			Pixel 2			...	Pixel 1024
	Color 1	Color 2	Color 3	Color 1	Color 2	Color 3	...	Color 3
m \ i	Input 1	Input 2	Input 3	Input 4	Input 5	Input 6	...	Input 3072
Image 1	I_{11}	I_{21}	I_{31}	I_{41}	I_{51}	I_{61}	...	$I_{3072\ 1}$

Matrix |O|

Output 1	Output 2	Output 3
O_{11}	O_{12}	O_{13}

Matrix|C|

	Input 1				Input 2				Input 3				Input 4			
	Intervals				Intervals				Intervals				Intervals			
	1	2	...	16	1	2	...	16	1	2	...	16	1	2	...	16
Output 1	$C_{1,1,1}$	$C_{1,2,1}$	...	$C_{1,16,1}$	$C_{2,1,1}$	$C_{2,2,1}$	...	$C_{2,16,1}$	$C_{3,1,1}$	$C_{3,2,1}$	...	$C_{3,16,1}$	$C_{4,1,1}$	$C_{4,2,1}$	...	$C_{4,16,1}$
Output 2	$C_{1,,12}$	$C_{1,2,2}$	...	$C_{1,16,2}$	$C_{2,1,2}$	$C_{2,2,2}$	...	$C_{2,16,2}$	$C_{3,1,2}$	$C_{3,2,2}$	...	$C_{3,16,2}$	$C_{4,1,2}$	$C_{4,2,2}$	...	$C_{4,16,2}$
Output 3	$C_{1,1,3}$	$C_{1,2,3}$	...	$C_{1,16,3}$	$C_{2,1,3}$	$C_{2,2,3}$	...	$C_{2,16,3}$	$C_{3,1,3}$	$C_{3,2,3}$	...	$C_{3,16,3}$	$C_{4,1,3}$	$C_{4,2,3}$	...	$C_{4,16,3}$
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
Output 10	$C_{1,1,10}$	$C_{1,2,10}$	...	$C_{1,16,10}$	$C_{2,1,10}$	$C_{2,2,3}$	...	$C_{2,16,10}$	$C_{3,1,10}$	$C_{3,2,10}$	...	$C_{3,16,10}$	$C_{4,1,10}$	$C_{4,2,10}$	...	$C_{4,16,10}$

Matrix |W|

	Intervals	Output 1	Output 2	Output 3
Input 1	1	$W_{1, 1, 1}$	$W_{1, 1, 2}$	$W_{1, 1, 3}$
	2	$W_{1, 2, 1}$	$W_{1, 2, 2}$	$W_{1, 2, 3}$
	3	$W_{1, 3, 1}$	$W_{1, 3, 2}$	$W_{1, 3, 3}$
	4	$W_{1, 4, 1}$	$W_{1, 4, 2}$	$W_{1, 4, 3}$
	5	$W_{1, 5, 1}$	$W_{1, 5, 2}$	$W_{1, 5, 3}$
	6	$W_{1, 6, 1}$	$W_{1, 6, 2}$	$W_{1, 6, 3}$
	7	$W_{1, 7, 1}$	$W_{1, 7, 2}$	$W_{1, 7, 3}$
	8	$W_{1, 8, 1}$	$W_{1, 8, 2}$	$W_{1, 8, 3}$
	9	$W_{1, 9, 1}$	$W_{1, 9, 2}$	$W_{1, 9, 3}$
	10	$W_{1, 10, 1}$	$W_{1, 10, 2}$	$W_{1, 10, 3}$
	11	$W_{1, 11, 1}$	$W_{1, 11, 2}$	$W_{1, 11, 3}$
	12	$W_{1, 12, 1}$	$W_{1, 12, 2}$	$W_{1, 12, 3}$
	13	$W_{1, 13, 1}$	$W_{1, 13, 2}$	$W_{1, 13, 3}$
	14	$W_{1, 14, 1}$	$W_{1, 14, 2}$	$W_{1, 14, 3}$
	15	$W_{1, 15, 1}$	$W_{1, 15, 2}$	$W_{1, 15, 3}$
	16	$W_{1, 16, 1}$	$W_{1, 16, 2}$	$W_{1, 16, 3}$
Input 2	1	$W_{2, 1, 1}$	$W_{2, 1, 2}$	$W_{2, 1, 3}$
	2	$W_{2, 2, 1}$	$W_{2, 2, 2}$	$W_{2, 2, 3}$
	3	$W_{2, 3, 1}$	$W_{2, 3, 2}$	$W_{2, 3, 3}$
	4	$W_{2,4, 1}$	$W_{2, 4, 2}$	$W_{2, 4, 3}$
...	...	...	...	...
Input 1024	1	$W_{1024, 1, 1}$	$W_{1024, 1, 2}$	$W_{1024, 1, 3}$
	2	$W_{1024, 2, 1}$	$W_{1024, 2, 2}$	$W_{1024, 2, 3}$
	3	$W_{1024, 3, 1}$	$W_{1024, 3, 2}$	$W_{1024, 3, 3}$
	4	$W_{1024,4, 1}$	$W_{1024, 4, 2}$	$W_{1024, 4, 3}$
	5	$W_{1024,5, 1}$	$W_{1024, 5, 2}$	$W_{1024, 5, 3}$

From the patent: US Patent US 10423694

NEURAL NETWORK AND METHOD OF NEURAL NETWORK TRAINING

PANN Batch Training

Real training begins with the process of recognition using steps 2 and 3 - building neural sums and subtracting these sums from the desired result to determine the error in recognition.

Step 2. Formation of the matrix of neural sums |∑|

Formation of the matrix of neural sums |∑| is produced by multiplying the value of each correction weight of the matrix |W| on the corresponding coefficient of influence from the matrix |C| and the subsequent addition of the values of all weights that affect the given neuron "n". This is done by matrix multiplication |C| and |W|.

$$|\sum w| = |C| \times |W| =$$

	Input 1				Input 2				Input 3				Input 4			
	Intervals				Intervals				Intervals				Intervals			
	1	2	...	16	1	2	...	16	1	2	...	16	1	2	...	16
Output 1	$C_{1,1,1}$	$C_{1,2,1}$	...	$C_{1,16,1}$	$C_{2,1,1}$	$C_{2,2,1}$	...	$C_{2,16,1}$	$C_{3,1,1}$	$C_{3,2,1}$	...	$C_{3,16,1}$	$C_{4,1,1}$	$C_{4,2,1}$	...	$C_{4,16,1}$
Output 2	$C_{1,,12}$	$C_{1,2,2}$	...	$C_{1,16,2}$	$C_{2,1,2}$	$C_{2,2,2}$	...	$C_{2,16,2}$	$C_{3,1,2}$	$C_{3,2,2}$	...	$C_{3,16,2}$	$C_{4,1,2}$	$C_{4,2,2}$	...	$C_{4,16,2}$
Output 3	$C_{1,1,3}$	$C_{1,2,3}$	...	$C_{1,16,3}$	$C_{2,1,3}$	$C_{2,2,3}$	...	$C_{2,16,3}$	$C_{3,1,3}$	$C_{3,2,3}$	...	$C_{3,16,3}$	$C_{4,1,3}$	$C_{4,2,3}$	...	$C_{4,16,3}$
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
Output 10	$C_{1,1,10}$	$C_{1,2,10}$	...	$C_{1,16,10}$	$C_{2,1,10}$	$C_{2,2,3}$	...	$C_{2,16,10}$	$C_{3,1,10}$	$C_{3,2,10}$	...	$C_{3,16,10}$	$C_{4,1,10}$	$C_{4,2,10}$	...	$C_{4,16,10}$

Matrix of coefficients of influence of corrective weights

$|C_{\text{input, interval, output}}|$

$\times$

	Intervals	Output 1	Output 2	Output 3
Input 1	1	$W_{1,1,1}$	$W_{1,1,2}$	$W_{1,1,3}$
	2	$W_{1,2,1}$	$W_{1,2,2}$	$W_{1,2,3}$
	3	$W_{1,3,1}$	$W_{1,3,2}$	$W_{1,3,3}$
	4	$W_{1,4,1}$	$W_{1,4,2}$	$W_{1,4,3}$
	5	$W_{1,5,1}$	$W_{1,5,2}$	$W_{1,5,3}$
	6	$W_{1,6,1}$	$W_{1,6,2}$	$W_{1,6,3}$
	7	$W_{1,7,1}$	$W_{1,7,2}$	$W_{1,7,3}$
	8	$W_{1,8,1}$	$W_{1,8,2}$	$W_{1,8,3}$
	9	$W_{1,9,1}$	$W_{1,9,2}$	$W_{1,9,3}$
	10	$W_{1,10,1}$	$W_{1,10,2}$	$W_{1,10,3}$
	11	$W_{1,11,1}$	$W_{1,11,2}$	$W_{1,11,3}$
	12	$W_{1,12,1}$	$W_{1,12,2}$	$W_{1,12,3}$
	13	$W_{1,13,1}$	$W_{1,13,2}$	$W_{1,13,3}$
	14	$W_{1,14,1}$	$W_{1,14,2}$	$W_{1,14,3}$
	15	$W_{1,15,1}$	$W_{1,15,2}$	$W_{1,15,3}$
	16	$W_{1,16,1}$	$W_{1,16,2}$	$W_{1,16,3}$
Input 2	1	$W_{2,1,1}$	$W_{2,1,2}$	$W_{2,1,3}$
	2	$W_{2,2,1}$	$W_{2,2,2}$	$W_{2,2,3}$
	3	$W_{2,3,1}$	$W_{2,3,2}$	$W_{2,3,3}$
	4	$W_{2,4,1}$	$W_{2,4,2}$	$W_{2,4,3}$
...	...	...	...	...
Input 1024	1	$W_{1024,1,1}$	$W_{1024,1,2}$	$W_{1024,1,3}$
	2	$W_{1024,2,1}$	$W_{1024,2,2}$	$W_{1024,2,3}$
	3	$W_{1024,3,1}$	$W_{1024,3,2}$	$W_{1024,3,3}$
	4	$W_{1024,4,1}$	$W_{1024,4,2}$	$W_{1024,4,3}$
	5	$W_{1024,5,1}$	$W_{1024,5,2}$	$W_{1024,5,3}$

Correction weight matrix

$|W| - |W_{\text{input, interval, output}}|$

PANN Batch Training

Step 3. Calculation of the error matrix of neural sums |E|. Error matrix **|E|** is formed by subtracting the neural sum matrix **|∑|** from the matrix of desired output signals **|O|**.

$$|E| = |O| - |\sum w|$$

Step 4. Calculate the correction for all weights that contribute weight to neural sums

$|\Delta W| = |E| / k$, where k is the number of pixels in the image

Step 5. Building the Adjusted Weights Matrix

$$|W_{n+1}| = |W_n| + |\Delta W|$$

Step 5. Training other images. Sequential repetition of steps 2 - 5 for all images. This ends the first epoch of training. Its result is a system of trained corrective weights $|W_{n+1}|$

Step 6. Reducing the error. If, after an epoch, the desired training accuracy is not achieved, several more epochs can be carried out by sequentially repeating steps 2–5 for all images until the training error becomes less than some predetermined value, which ends training.

Note. Training according to this scheme combines training as such and the normalization of weights, which is necessary to ensure the comparability of different images.

Recognition in a trained PANN network

The recognition process is extremely simple:

Step 1. Formation of the matrix of the recognizable input image **| Ir | (**as shown on slide 14).

Step 2. Formation of the matrix of coefficients of influence of corrective weights **|Cr| (**as shown on slide 14).

Step 3. Formation of the matrix of neural sums **|∑r|** by multiplying the matrix **|Cr|** of a recognizable image
and the matrix of trained weights **|Wt| (**as shown on slide 15):

$$|\sum r| = |Cr| \times |Wt|$$

Step 4. Each local (on one neuron) neural sum $\sum_1, \sum_2, \sum_3, \ldots$ from the matrix of neural sums **|∑r|** represents the degree of similarity between the recognizable image **| Ir |** and different training images as a Pearson correlation coefficient.

The simplicity of the recognition process requiring a minimal number of mathematical operations (a single multiplication of two matrices takes just milliseconds) can be essential for using the PANN network in real-time, for example, when controlling drones, uncrewed vehicles, etc.

PANN Training Management System

According to the disclosure, a “progressive artificial neural network” has a plurality of synapses and a plurality of corrective weights associated with each synapse.

The training method includes receiving, via the plurality of inputs 102, the training images 106. The training image 106 may either be received as the training input value array 107 prior to commencement of the subject training phase or codified as the training input value array during the actual training phase. The method includes organizing the corrective weights 112 of the plurality of synapses 118 in the corrective weight array 119. Each synapse 118 is connected to one of the plurality of inputs 102 and includes a plurality of corrective weights 112.

The method includes receiving, via the controller 122 and a data processor 150, desired images 124 organized as the desired output value array 126 and determining, via the controller 122, the deviation 128 of the neuron sum array 120 from the desired output value array 126 and thereby generate the deviation array 132.

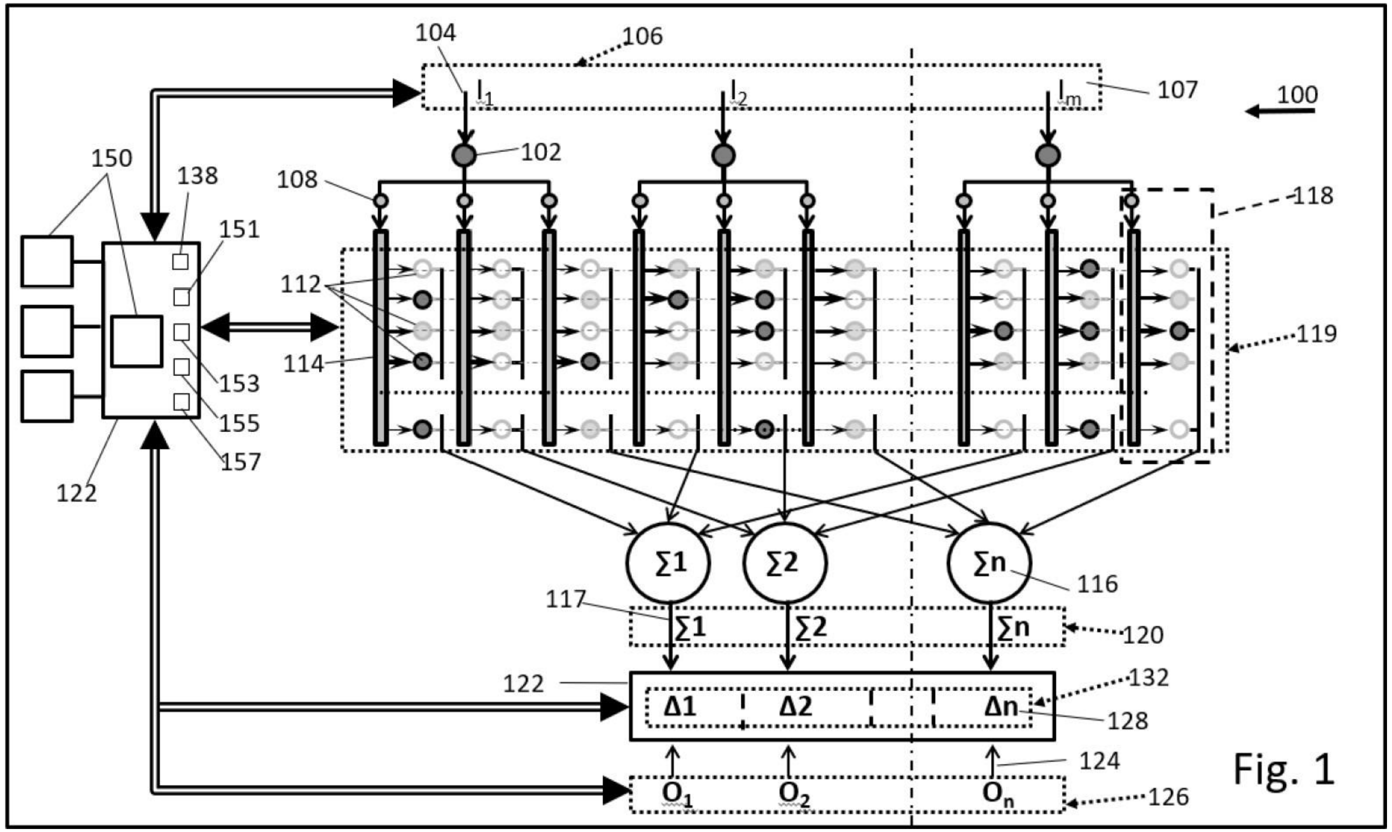
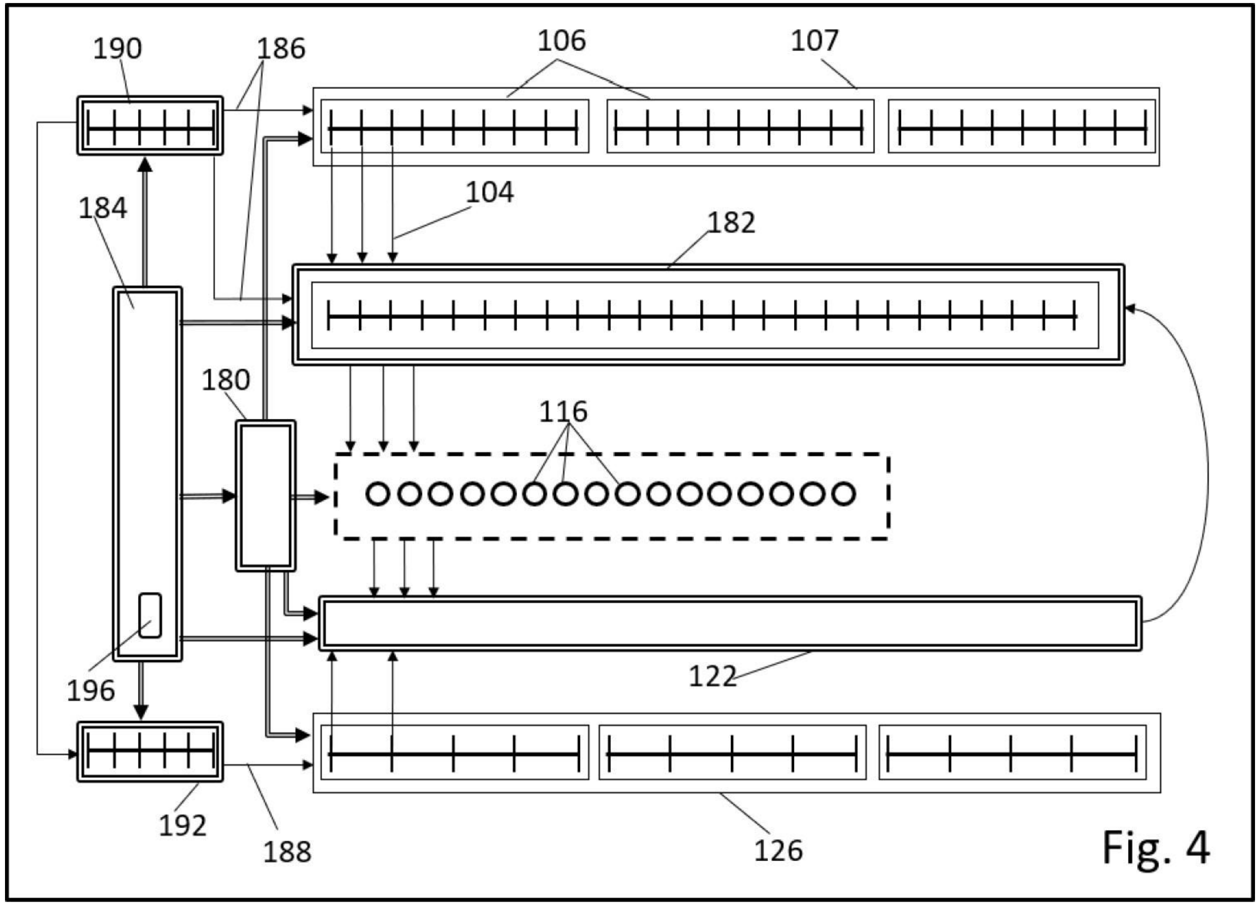


Fig. 1

From **Patent Application**
NEURAL NETWORK AND METHOD OF NEURAL NETWORK TRAINING
Notice of Patent Allowance 16/523.584; 06.29.2022

PANN Training Management



106 – Input images	180 – Signal distribution system
107 - Input signals array	182 - Shared memory
116 - Neuron units	184 – Structure forming system
122 - Weight correction calculator	190 – Array of testing images
126 - Desired output signals array	

A method of training the PANN has the operative structure including the structure-forming module 180, the signal allocation module 182, and the training module 184. The method is configured to enhance the training of the p-net when operated on an apparatus, such as a computer or a system of computers employed in implementing supervised training using one or more data processors. The method may be programmed into a non-transitory computer-readable storage device for operating the p-net and encoded with instructions executable to perform the method.

From **Patent Application**
NEURAL NETWORK AND METHOD OF NEURAL NETWORK TRAINING
Notice of Patent Allowance 16/523.584; 06.29.2022

PANN Batch Training with generalization, clustering and classification

The disadvantage of the described PANN training and recognition algorithms is the sizeable neural network size due to the recording of each trained image. It is possible to significantly reduce the required memory by dividing the training package into a set of classes, in which the closest, similar images (for example, photographs of one person) are assigned to one class. In most practical cases, this allows for an increase in the speed of both training and recognition.

In PANN, it is possible to represent each class in a generalized way. This is achieved by training one selected “class neuron” with the entire set of images assigned to this class.

PANN implements a "double recognition" algorithm, in which each image is recognized twice:

1. As a representative of a particular class
2. As a member of some dynamically formed “high correlation” group that could be considered as a new class if necessary

This makes it possible to implement Explanatory Artificial Intelligence, which is essential for medical and many other applications.

Selected Examples of PANN Applications

In this presentation (as in books and articles on neural networks), the word "image" reference is not limited to "pictures" but rather to any object presented for recognition in the form of a numerical vector. For example, an image can be a table, a graph of a curve in numerical form, a set of experimental points, visual or sound spectra, etc.

The following slides show selected specific examples of how **PANN** works:

- Slide 22 presents the **PANN VISUAL IMAGE MATRIX (PVIM)** software, designed to recognize visual individual and generalized images (pictures, photographs, films, etc.) and compare them by correlation coefficients, clustering, and classification. In this example, the **PVIM** software recognizes lung diseases from X-ray images.
- Slide 23 presents the **PANN TABLE MATRIX (PTM)** software designed to recognize numerical sequences in tabular form and compare them by correlation coefficients, clustering, and classification. The PTM software is used for chronic heart failure diagnostics based on daily body weight changes in this example.
- Slide 24 shows the result obtained by the **PTM** software - a risk graph and a visualization of "risk days" for a person diagnosed with chronic heart failure.

Both software are ready for practical use, conditional on the availability of appropriate libraries for training.

TFC2022

Software PANN VISUAL IMAGE MATRIX for Lung Diagnostics

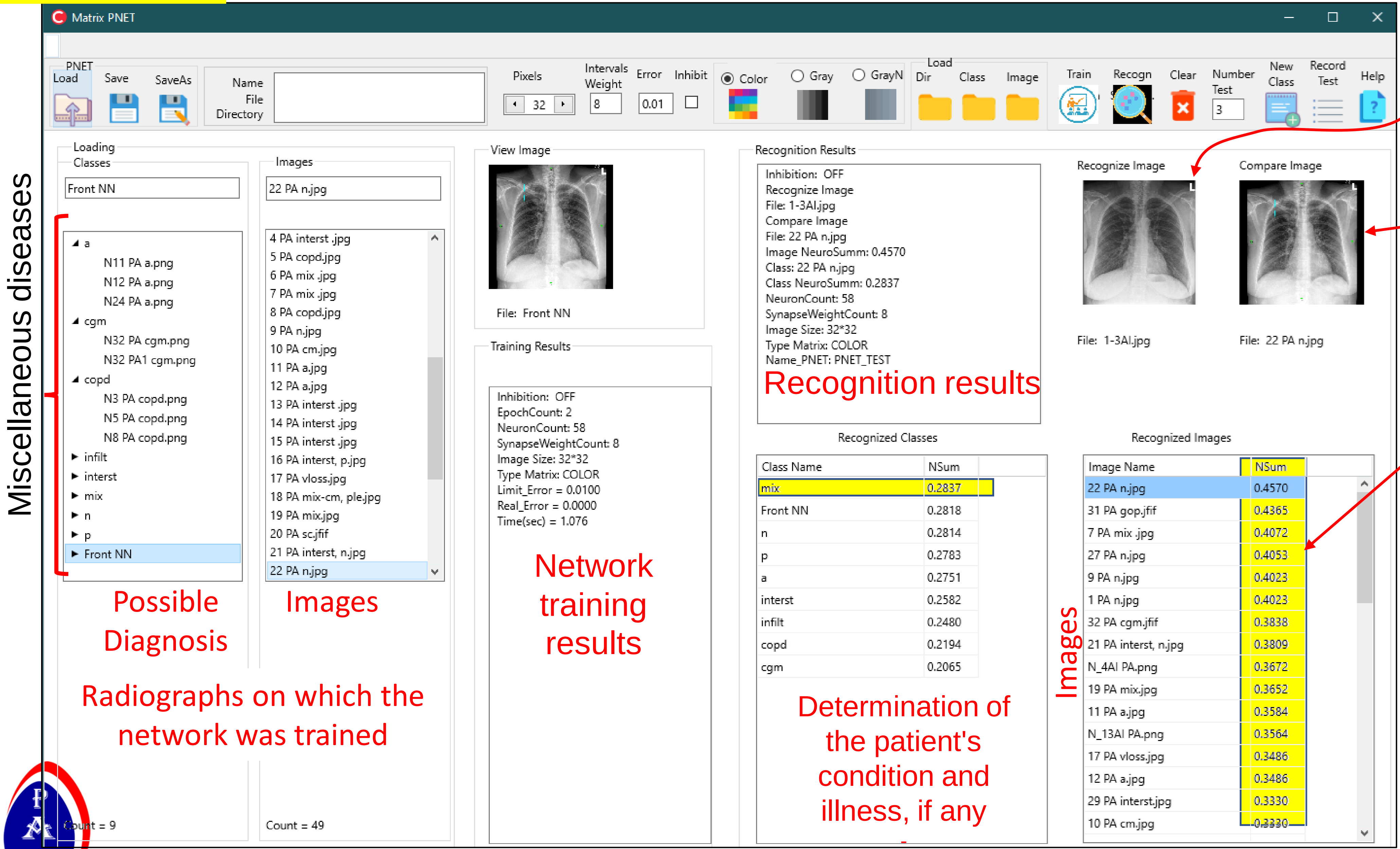


Image in question

The image from those used in training that is closest to the one under consideration

The correlation coefficient (degree of similarity) of the considered image with those used in training

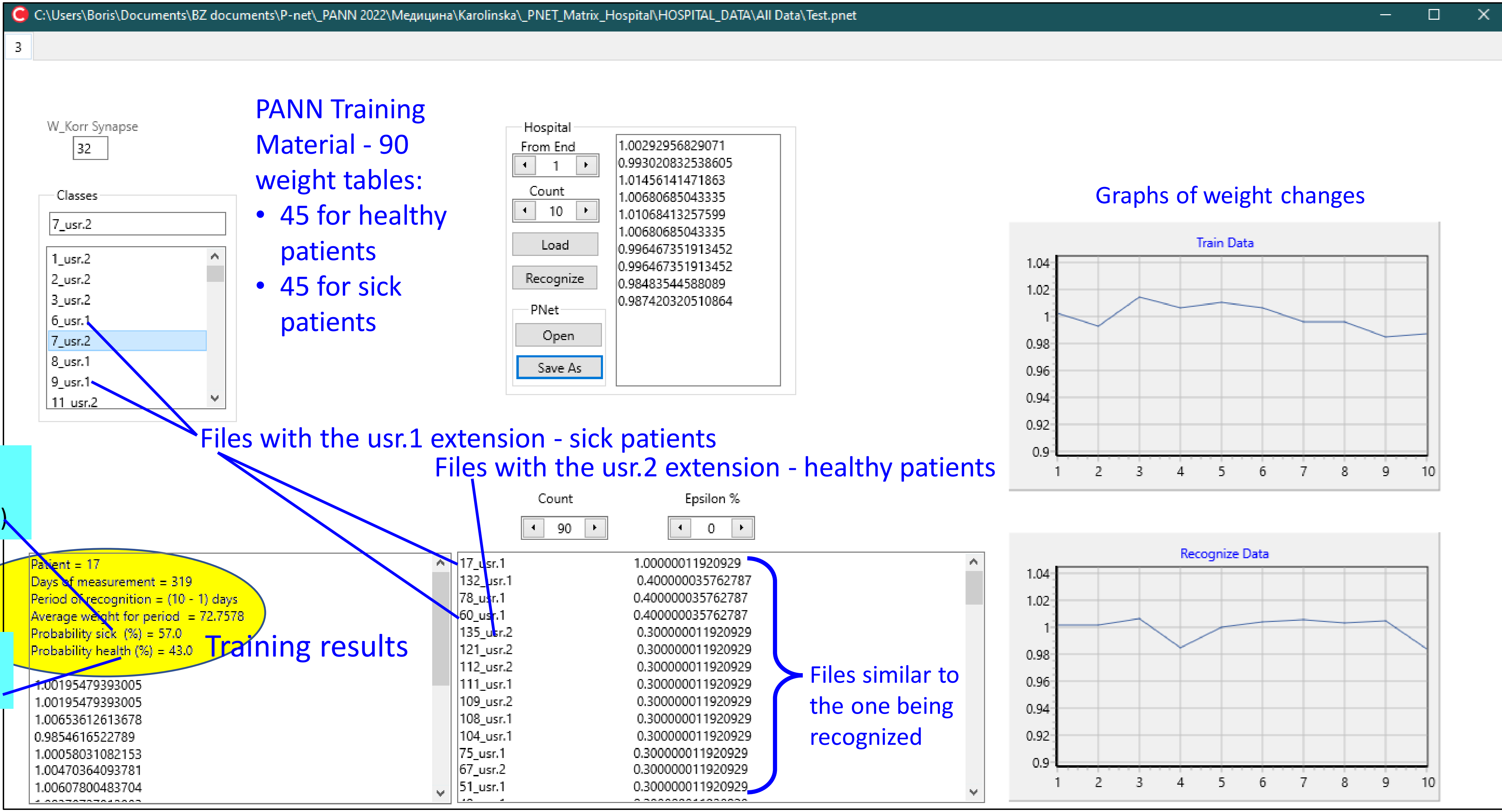
Software PANN Table MATRIX applied in the realization of the Algorithm for chronic heart failure diagnostics based on daily body weight

Probability of sick =

$\sum_1^{90} 1 \,/\, (\sum_1^{90} 1 + \sum_1^{90} 2\,)$

Probability of healthy =

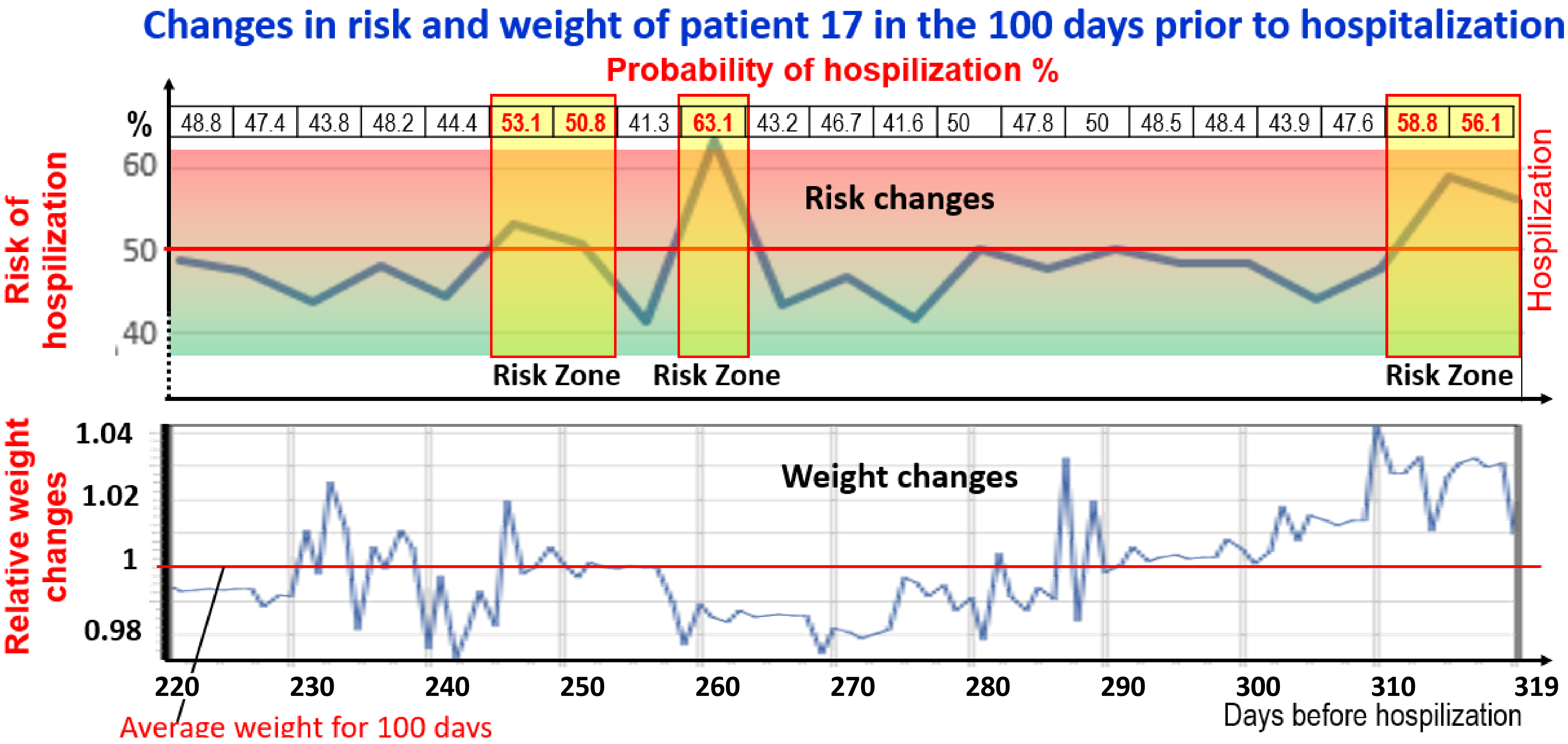
$\sum_1^{90} 2 \,/\, (\sum_1^{90} 1 + \sum_1^{90} 2\,)$



Demonstration of typical PANN recognition algorithm

Chart of weight changes for patient 17 for the last 100 days prior to the second hospitalization and the risk degree in various moments. There was a real danger of heart failure during the period between day 75 and day 55 before hospitalization. There was 50-day "remission" afterward, followed by the last 10 days of risks increased, ending with hospitalization.

It means that in the process of monitoring, a dangerous period was missed, and then for the last 50 days the patient was close to a danger



Health networks

It was found that **PANN_Table** allows determining with high-reliability periods of increased risk of relapses and exacerbations of the disease, which conventional methods do not notice. That is, such software, the small size of which allows it to be installed, for example, on a telephone associated with simple electronic scales, can predict the risk of heart failure based on the results of daily weight measurements.

Also, other characteristics of the organism can be used as “danger predictors,” for example, data on changes in temperature, blood pressure, pulse, blood oxygen saturation, cardiograms, noises of different ranges in the lungs, arteries, and joints, mechanical and electrical characteristics of the skin, muscles, active points, chemical properties of exhalation, sweat and metabolic products, echogram data, etc. And, of course, from this data, heart problems can be diagnosed, but also, after appropriate training of the PANN, the risks of other diseases, such as diabetes, hypertension, metabolic disorders, autoimmune diseases, and even, apparently, cancer and COVID.

PANN can complete training in the operation process and thus adapt to a specific user. It can gradually convert PANN into a **“Personal Medical Adviser”** or **“Personal Health Network”** that will determine the risks and make recommendations based on the characteristics of a particular user. Another essential function is issuing the necessary, processed information to the attending physician if necessary.

Set of **“Personal Health Network” Sets of “Personal Health Networks”** can be linked to higher-level networks, such as the health networks of an insurance company, a district or city, and so on. This will allow us to upgrade the software and recognize the epidemic’s beginning, collect information about the effectiveness of specific therapeutic agents and recommendations, etc.

Selected general results of the PANN testing

The experiments confirmed our expectations of a drastic acceleration of network training by eliminating iterative calculations and radically reducing the required number of training epochs. In most cases, one epoch was sufficient.

The tests also revealed several additional positive findings:

- PANN network has proven unnecessary the use of the activation function typical for classical neural networks. Abandoning this function allowed for an additional increase (several times) in the speed and accuracy of training.
- PANN network allows for additional training without losing the information of the previous training. Moreover, additional training is fast and can be conducted in real-time.
- The possibility of building and visualizing "generalized images" and using them to classify groups of images.
- PANN can be implemented and trained on simple equipment, up to laptops and smartphones, imposing fewer requirements on training material preparation.
- , Unlike classical neural networks, PANN is fully scalable, making it possible to increase the information capacity of the network by adding new neurons when it is reduced or exhausted.
- PANN has proven to be less prone to paralysis, freezing, and overfitting than classical neural networks. In our experiments, these effects have never been produced.

Possible PANN Applications

The use of PANN is possible primarily in the field of information technology, in particular:

- Hybridization with classical neural networks, allowing for effective use of the essential materials and knowledge bases developed in the field of neuroscience for over 60 years.
- Using trained PANNs as objects for Object Oriented Programming Languages.
- Creation of Smart Database Management System (SDBMS) compatible with existing DBMS and complementing them with specialized intellectual functions, for example, the possibility of fast associative search, pattern recognition, automatic classification, etc.
- Application of PANN to emulate various software products
- Improving computer security systems based on fast training PANN
- Creation of highly efficient specialized AI systems for practical applications

PANN can be used as a consumer product in all areas of human activity where the availability of inexpensive Artificial Intelligence can be critical, in particular:

- Built-in “intelligent controller” for technical devices, appliances, and other consumer products
- Personal teacher and trainer, translator, negotiator, adviser, accountant, computer-Internet agent, constantly collecting valuable information to the owner, etc. - a lifelong person’s companion as "Alter Ego.”
- A personal doctor who monitors a person’s health and emotional state, a protector capable of foreseeing dangers and helping prevent them.

Applying PANN to TRIZ

One of the laws of system development discovered over the past decades in the framework of I-TRIZ is

the Pattern of Unlimited Development:

- Any invention is the result of a particular resource presence
- Any invention generates new resources, which, accordingly, generate new inventions.

The creation and development of PANN became possible thanks to such a powerful tool of creativity and systems thinking as TRIZ. And now, PANN is becoming a new resource that can significantly improve TRIZ.

Over the past 30 years, Ideation International has developed a set of I-TRIZ software, which are programs with elements of algorithmic Artificial Intelligence, which are very effective, but mastering them is pretty tricky. Today there is a theoretical and experimental development of a new generation of TRIZ software with enhanced intelligence and self-adaptation capabilities for the user through PANN. These programs will be much more user-friendly; we believe that in alliance with PANN, TRIZ should become part of the mass and everyday skills of the average person.

Applying PANN to TRIZ

We are specialists in the Theory of Inventive Problem Solving with rich experience in developing software products based on I-TRIZ. And over the past ten years, we've also become PANN specialists. We argue that the already achieved level of development of PANN is sufficient to create systems for supporting creativity, operating in a hybrid human-machine mode. For example:

- Using PANN for associative search in professional databases and on the Internet
- Using PANN machine intelligence in the formation and analysis of cause-and-effect diagrams, automatic formulation of tasks and contradictions
- Teaching PANN to search for inventive resources in systems and select operators for solving innovative problems corresponding to a given problem
- Teaching PANN application to the formulated problem of operators in solving problems, as well as the laws and lines of development of systems in forecasting

Another promising possibility is using PANN for an interactive TRIZ self-training system that adapts to a person. In particular, systems for teaching creativity and developing the creative imagination of children and adolescents.

TRIZ + PANN tomorrow

In the 19th and 20th centuries, in the human environment, there was a rapid replacement of devices and processes that use human energy for functioning with systems that use external energy sources - steam, electric, internal combustion systems, etc.

Today, a similar process is gradually gaining momentum - the penetration of an increasing number of intelligent products into our environment. So far, this process is hampered by the high cost of developing classical neural networks and the need to use high-end computers with colossal power consumption.

PANN allows hundreds and thousands of times to reduce the cost of developing intellectual products, the necessary equipment, and energy costs. This radically changes the possibilities for the business of neural networks and the future of artificial intelligence. The development and creation of intelligent products by small companies and even at home is becoming real, as, in the nineties, the first Internet products were developed.

At the same time, the biggest problem will be the need to invent a mass of new intelligent products unknown to us so far. This should significantly increase the still weak demand for TRIZ.
We expect the near future to create businesses based on
hybridization of TRIZ and PANN for complex service of innovative product development.

PANN + I-TRIZ in Omega Server

1. ***PANN creation and training for special applications.*** Possibility to develop effective intelligent products quickly, with low money and human labor costs. It is also possible to jointly carry out some projects.
2. ***The invention of intellectual products for implementation using PANN,*** based on the latest modification of the Theory of Inventive Problem Solving (I-TRIZ) and software products based on it, in particular, software for the invention of new products and processes Innovation WorkBench - Omega (IWB-Ω).
3. ***Finance.*** The Omega Server platform will function as a "Software as a Service" (SaaS) and facilitate the creation and prosperity of startups, including their financing through crowdsourcing and protection using PANN, I-TRIZ, and blockchain technologies.
4. ***Intellectualization tornado coordination*** standardizes the characteristics of different products based on PANN for their integration and complex use by customers.
5. ***Education*** prepares individuals to create and use products based on PANN and I-TRIZ and live in the new world of intelligent products and services.