and non-directional training algorithms. This framework significantly limits the class of constructed neural networks and has all the disadvantages of gradient methods.

Thus, today the following disadvantages are typical for modern frameworks.

Most of them either have a limited set of training algorithms, or support a limited neural networks class, or have low performance.

III. Framework architecture and structure

Consider a variant of a software package implemented in a form of framework in which random search algorithms are used to train neural networks.

The software package was developed in C++ using the OpenMP and OpenCL libraries. OpenMP libraries provide efficient organization of parallel computing within a single processor, while OpenCL provides compatibility and parallel computing on a wide class of computing devices.

According to the ostis 2021 standard, the framework can be described as follows:

framework
⇒ *decomposition∗* :
{ • *algorithms library*
• *train behaviour parameters*
• *architecture library*
• *database*
• *compress/decompress module*
• *predict module*
• *load/save module*
}

algorithms library
$\ni FAST\_ANNEALING$
$\ni SLOW\_ANNEALING$
$\ni GENETIC$
$\ni SGD$
$\ni MOMGRAD$
$\ni ADAM$
$\ni FTML$

train behaviour parameters
$\ni NO\_TRAIN$
$\ni JUST\_TRAIN$
$\ni NEW\_TRAIN$
$\ni CONTINUE\_TRAIN$

architecture library
$\ni RBM\_BERNOULLI\_BERNOULLI$
$\ni RBM\_GAUSS\_BERNOULLI$
$\ni AUTOENCODER$
$\ni PERCEPTRON\_NN$
$\ni CONV\_LAYER$
$\ni POOLING\_LAYER$

PERCEPTRON_NN
⇒ *part*:*
ActivationFunction

AUTOENCODER
⇒ *part*:*
ActivationFunction

ActivationFunction
$\ni NONE$
$\ni BIPOLYARSIGM$
$\ni SIGM$
$\ni ReLU$
$\ni SOFTMAX$

The developed software package consists of the following main modules: two libraries (algorithms and architectures of neural networks), a database and a database with configuration files (for setting up algorithms from the library), modules (for execution on connected computing devices and the neural networks functioning), and finally, user interaction interface.

The database contains all the necessary data sets for training neural networks. The data is loaded at the user request. The framework has built-in methods for generating various types of samples (from the requested data) for training and testing neural networks.

The algorithm library contains a wide range of different optimization algorithms: simple gradient, moment and adaptive moment methods, following the moving leader method, genetic algorithm and annealing method. Configuration files are loaded during the framework initialization and contain sets with optimal parameter values, which, if necessary, are recalculated taking into account the neural network architecture. Execution Modules on connected computing devices are loaded as they are found. The framework also has a high degree of flexibility, it can be executed on a limited number of processor threads (the limit can be adjusted), on several connected computing devices. In addition, a completely single-threaded execution mode is possible.

The framework allows assembly without the use of parallel computing on connected devices and the processor. For this, the constants.h file contains the DISABLE_OPEN_CL, DISABLE_OPENMP constants. When set to non-zero values, the framework disables the ability to use parallel computing. This allows the framework to be independent of the settings and computer configuration. The framework has additional constants ENABLE_FAST_MATH and ENABLE_NORMAL_DISTR. They allow us to activate the possibility of using accelerated trigonometric functions, for example, to generate a normal distribution. They also activate the tabled