CJES Caspian Journal of Environmental Sciences
Online ISSN: 1735-3866
Print ISSN: 1735-3033

Development of an application for creation and learning of neural networks to utilize in environmental sciences

Ilgiz Rustamovich Sultanbekov*, Irina Yurievna Myshkina, Larisa Yurievna Gruditsyna

Department of Information Technologies and Energy Systems, Naberezhnye Chelny Institute (branch) of FSAEI HE KFU, Russia

** Corresponding author's E-mail henkok398@mail.ru*

ABSTRACT
Machine learning methods originated from artificial intelligence and today are applied in several fields concerning environmental sciences. Thanks to their powerful nonlinear modelling capability, machine learning methods today are utilized in satellite data processing, general circulation models(GCM), weather and climate prediction, air quality forecasting, analysis and modelling of environmental data, oceanographic and hydrological forecasting, ecological modelling, and monitoring of snow, ice and forests. Currently, the popularity of neural networks is growing; their areas of application are constantly expanding. In these conditions, the task of choosing a convenient tool for utilizing in environmental science with neural networks becomes urgent. There are many tools for working with neural networks, but each of them has its own drawbacks. So most of the existing tools require users to have programming knowledge; there are no tools to help quickly select the optimal network structure for the problem being solved. The purpose of the research is to simplify the process of choosing the optimal structure of an artificial neural network by developing an application with a graphical user interface with a visual representation of the stages of creating and learning neural networks in environmental sciences. The object of research is artificial feed-forward neural networks. Research work on the study, comparison and analysis of existing tools for the creation, learning and use of artificial neural networks has been carried out. Based on the research results, an application with a graphical interface aimed at solving the assigned tasks has been developed. An application developed to achieve this goal works correctly, without failures, and allows creating and learning feed-forward neural networks without programming knowledge.

Keywords: Machine learning, Artificial neural networks, Gradient descent, Qt, Clean architecture, Environmental science.

INTRODUCTION
Currently, artificial neural networks (ANNs) are used to solve many complex analytical problems, such as classification, prediction, and recognition (Rojas 1996; Goodfellow *et al*. 2016). ANN has gained particular popularity among specialists from semi-structured subject areas such as medicine, ecology, sociology, finance, and other practically important areas (Myshkina *et al.* 2015; Asanov & Myshkina 2017; Zhang *et al*. 2020). This is due to the fact that ANN systems do not need a previously known model to build it, but make it themselves on the basis of the information presented. That is why neural networks have come into practice wherever there are problems poorly compliant with algorithms. However, the use of such a powerful tool requires a person to understand how it works thoroughly. Unfortunately, ANNs are quite complex, and many face difficulties already at the initial stages of the study. This is due to the fact that at the moment there are no exact algorithms for constructing the ANN architecture and choosing the optimal learning rate, which, moreover, also vary depending on the task at hand. However, until the greatest minds of humankind have come up with such an algorithm, tools are required that can facilitate the choice of the ANN architecture and its learning parameters.
The purpose of this work is to simplify the process of choosing the optimal structure of an ANN by developing an application with a graphical user interface having a visual representation of the stages for creating and learning an ANN. The application is intended, among other things, for users with minimal knowledge in the field of machine learning, the application allows learning ANNs providing the ability to monitor this process. Due to its high popularity, a lot of convenient tools have appeared that allow creating, learning and using ANNs to solve user problems. Every year these tools become more convenient; new versions of existing ones, as well as completely new software products, are released. Let's consider the most famous of them.

Caspian J. Environ. Sci. Vol. 18 No. 5 (Special Issue: Environmental Aspects of Economic and Social Sustainable Development)
pp. 595~601 Received: April 29. 2020 Accepted: Sep. 06. 2020
DOI: Article type: Research

Monte Carlo tree search with an extension strategy network that controls the pursuit, and a channel organization to pre-select the most encouraging retrosynthetic steps were consolidated. These profound neural organizations were prepared on basically all responses ever distributed in natural science. This framework tackles for twice the same number of atoms, multiple times quicker than the customary PC helped search strategy, which depends on separated standards and hand-planned heuristics. In a twofold visually impaired AB test, scientific experts on normal considered the PC created courses to be equal to announced writing courses (Segler *et al.* 2018). It is shown that miserly profound learning models can conjecture a mind boggling nature of a transient precipitation field advancement and vie for the cutting edge execution with grounded nowcasting models dependent on optical stream procedures (Ayzel G *et al*. 2019). The capacity to learn undertakings in a consecutive design is essential to the improvement of man-made consciousness. As of not long ago neural organizations have not been equipped for this and it has been broadly imagined that disastrous failing to remember is an inescapable component of connectionist models. it is indicated that it is conceivable to beat this restriction and train networks that can keep up mastery on undertakings that they have not experienced for quite a while (Kirkpatrick *et al*. 2017). TensorFlow is an open-source software library for deep machine learning.

It includes a flexible set of tools that allows us to use the most advanced machine learning technologies. With its simple dependencies, TensorFlow can be quickly deployed across multiple architectures (Rasooli *et al*. 2018; Shakla 2019). Aftereffects of Poison relapse work affirmed that utilization of defensive garments, goggles, cover, gloves and boots during pesticide splashing altogether diminished human wellbeing presentations. It is prescribed to apply the amount of supplements and synthetic compounds recommended by ANN technique (Elahi *et al.* 2019). Neural Network Toolbox is a set of MATLAB tools that includes more than 150 different functions, which are divided into several groups, providing a user with ample opportunities for creating, learning, and using various ANNs (Nikolaeva 2015). Thanks to this, the user can solve problems of different types. Automated Neural Networks is a set of tools included in STATISTICA. It allows us to design our own networks using a graphical interface. The user interface is fully translated into Russian. NeuroSolutions is software that allows us to quickly and effortlessly create and train an ANN.

The general pros and cons of the considered software products are shown in Table 1. After considering the most famous analogs of the application developed in this work, we can come to the conclusion that programming knowledge is required to create our own ANN architecture.

This is due to the fact that applications with a graphical interface do not allow us to choose the required architecture and create it automatically independently. In addition, only one of them has support for the Russian language, without which it is quite difficult to use all the available functionality of the application correctly. Thus, the development of an application with a graphical user interface for creating, learning and using ANNs with the ability to choose their own architecture of the developed network is still an urgent task. The ability to monitor the learning process will allow us to fine-tune the learning parameters.

Table 1. Pros and cons of the considered software products.

	TensorFlow	**Neural Network Toolbox MATLAB**	**Automated Neural Networks STATISTICA**	**Neuro Solutions**
Availability of a graphical interface	–	+	+	+
Choice of ANN architecture	+	–	–	–
No programming knowledge required	–	+	+	+
Russian language support	–	–	+	–
Availability in the public domain	+	–	–	–

MATERIALS AND METHODS

Requirements for the developed application

The main interface of an application is a graphical tool for designing ANN architectures and their further use. The application should allow the user to create the desired architecture by adding, moving, deleting and connecting various units in the graphics scene. Each unit performs a specific function, such as a neuron. A unit on the graphics scene is depicted as a geometric fig. (for example, a circle is a neuron, and a square is a convolutional neuron).

Primary requirements:

There should be tools that make it easy to add, move, delete and connect different units using arrows, which we will traditionally call synapses.

The connection of units should be depicted as an arrow indicating the direction of data flow from one unit to another. It should also be possible to remove links.

It should be possible to configure the parameters of each unit, for example, the selection of the activation function used.

It should be possible to select the parameters of learning the ANN.
It should be possible to save and load the created architecture for later use, including all the parameters required for learning and using the ANN.
The graphics scene should allow the user to scale and move the work area with the ability to reset to its original state.
The scene should enable the user to observe the learning process of the ANN in real time.
The user should be able to turn off visualization as needed to speed up the learning process.
The architecture of the application should make it easy to add new functionality, for example, adding a new unit type.
The application must be cross-platform with the possibility of further transferring it to mobile devices.

Development of algorithms for data feed-forward and backpropagation

Since the application must allow dynamically constructing various ANN architectures using special units, it is required to develop unique algorithms for data feed-forward and backpropagation. The fact is that in the developed application there is no possibility of combining neurons into a layer; in all existing algorithms, calculations proceed from layer to layer and the computation data of the previous layer is needed to compute the current layer. For a complete understanding of the problem, consider a specific example. The user has almost finished the architecture development, but at the last moment, it was realized that he was missing one more neuron for the input data.
He adds another one without any problems and sets up all the links. But from the software side, this neuron was added last; the program does not know that it needs to compute this neuron along with all the neurons of the input layer and will compute it as the last one. This will lead to incorrect computations of the entire neural network. In this connection, the task was to develop an algorithm that allows each neuron to compute correctly without knowing that it belongs to a certain layer.
To solve the problem, it was decided to distinguish three central units of computation: the input data unit, the neuron unit, and the synapse connecting the units. Each unit only knows about its input and output synapses. And synapses only know about their input and output units. The computation starts with a unit of input data. The rest of the units begin computations only after receiving a signal about data readiness from all input synapses, during forwarding propagation of data, or from all output synapses during backpropagation. After computation, each unit sends the results to all input or output synapses, depending on the kind of data direction. Thus, the computation goes through all neurons until it reaches the output neurons. In feed-forwarding, output neurons transmit their computations to an input unit via output synapses that are input to an input unit. This concludes the feed-forward algorithm.
Backpropagation of an error also begins from a unit of input data, which receives the computation results of the output neurons and, based on them, computes the total iteration error and errors of each neuron. The computed errors are sent to the corresponding neuron of the output neuron. Then the error is transmitted from the output layer to the input layer in the same way as with feed-forwarding. Thus, the backpropagation algorithm ends at the input neurons.

Application of architecture design

The clean architecture was used as the basis for designing the application architecture. Clean architecture is a set of guidelines for building applications that were described by Robert Martin (Martin 2019).
The clean architecture combines the ideas of several other architectural approaches that agree that architecture should have the following characteristics:
Independence from software libraries - the architecture does not depend on the availability of any library;
Ease of testing - the main logic of the application can be tested without a graphical interface;
Independence from the user interface - the graphical interface can be easily changed without affecting the rest of the system.
This is achieved by dividing the architecture into layers and following the dependency rule. Dependency rule is a code that implements the main logic of the application; it should not depend on the code that implements the interaction with the graphical interface or external program libraries (Martin 2019). This rule allows us to design application architectures that will be easier to maintain in the future.
There are three main layers in clean architecture:
Presentation Layer - contains the code that implements the graphical user interface and interaction with it;
Domain Layer - contains the code that implements the main logic of the application;
Data Layer - contains the code that implements loading and saving data.
Each layer is developed as a separate module, which will allow us to replace or use the developed modules in other applications. Now, having defined the general requirements for the developed application, we can create its general architecture (Fig. 1).
Dashed arrows represent usage relationships. Arrows with a triangular head correspond to an implementation or inheritance relationship.

It is important to note that the arrows from the data management module and from the presentation module are directed towards the main module. This means that the main module is independent of other modules. Thus, changes in the data module or in the presentation module will not affect the main module.

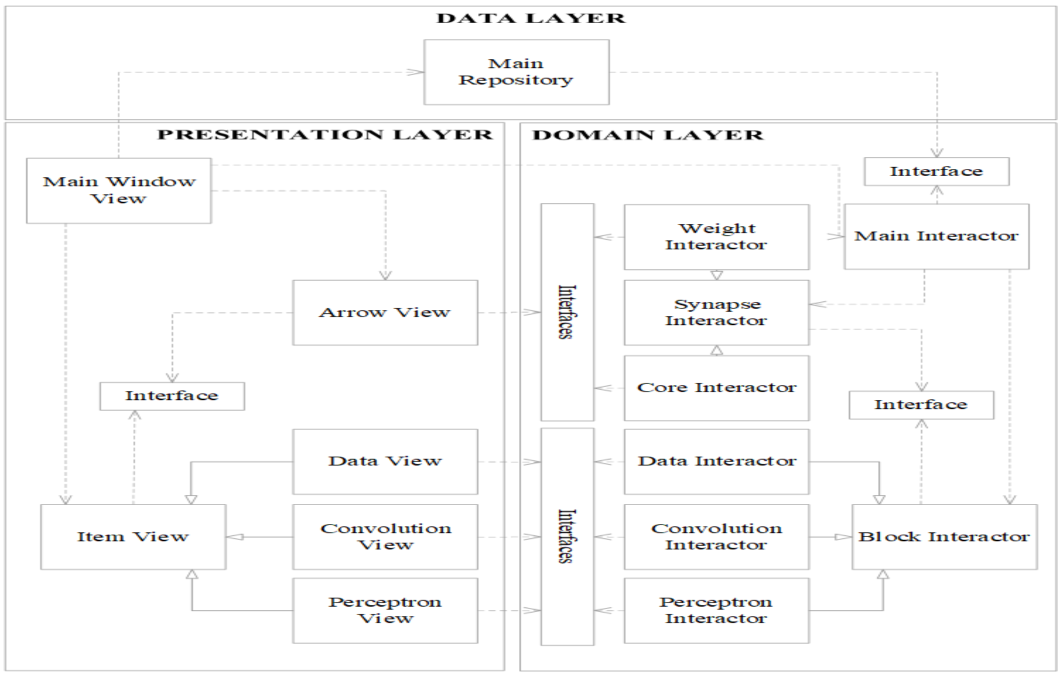

Fig. 1. The general architecture of the developed application.

Choice of the software environment

C ++ was chosen as the programming language. It is an object-oriented low-level programming language. Object-orientedness of C ++ means that it supports a programming style that makes it easy to code large-scale programs and ensures their extensibility (Stroustrup 2000), which is one of the requirements for the developed application. As a low-level language, C ++ can generate highly efficient high-speed programs (Stroustrup 2000). This is also an essential criterion for applications working with ANN, since learning a network can take a huge amount of time, which must be reduced by any means. Qt Creator was chosen as the development environment designed to work with the Qt framework. Its main feature is that it allows us to develop one application for different platforms using common development and debugging tools. Qt provides support for a wide range of operating systems: Microsoft Windows, macOS X, Linux, as well as mobile operating systems iOS, Android, Windows Phone, Windows RT and BlackBerry (Schlee 2018).
The provided plug-in system allows us to create modules that extend the functionality of the created applications (Schlee 2018). This functionality of the selected development environment will allow us to implement each module separately with the possibility of their further support or complete replacement with a new one. Qt also contains many tools for creating and optimizing a GUI.

Development of the Domain module

The Domain module is the foundation of the entire application. It implements all the logic of learning and using the ANN. It manages the learning process, and its visualization provides interaction between the presentation and data management modules. It is with the help of this module that all elements of the graphic scene are added, edited and deleted.
This module is implemented in the C ++ language without any external libraries, which will allow us to use this module in the future in different development environments besides Qt. Special classes called interfaces are used to interact with the presentation module and the data management module. An interface specifies a list of methods that must be implemented by every class that implements the interface. Thanks to interfaces, the main module will be protected from changes in external modules, thereby simplifying the development of the application.

Development of the presentation module

The presentation module is responsible for the graphical interface of the application. This module implements a visualization of data received from the main module. Also, the module is responsible for handling events triggered by the user and informs the main module about the changes, depending on the type of event. The graphical interface in Qt consists of so-called widgets. A widget is not just an area displayed on the screen; it is a component capable of performing various actions, for example, reacting to incoming signals and events or sending signals to other widgets (Schlee 2018). The graphical interface is a window that hosts the MainWindow widget. Interaction with the domain module is carried

Received: April 29. 2020 Accepted: Sep. 06. 2020
DOI:
Article type: Research

out using its interfaces. Thanks to this, this module can be replaced with other graphical interfaces, including those that were developed for other platforms.
The process of interaction between modules by the example of a unit of input data is shown in Fig. 2. The double line shows the border of the modules. The fig. shows that thanks to the Data Presentor Listener and Data Interactor Listener interfaces, all arrows are directed towards the domain module. The interaction with the rest of the units and the graphical scene itself with the domain module takes place in the same way. Thus, the domain module does not need to know with which presentation module it is currently interacting.

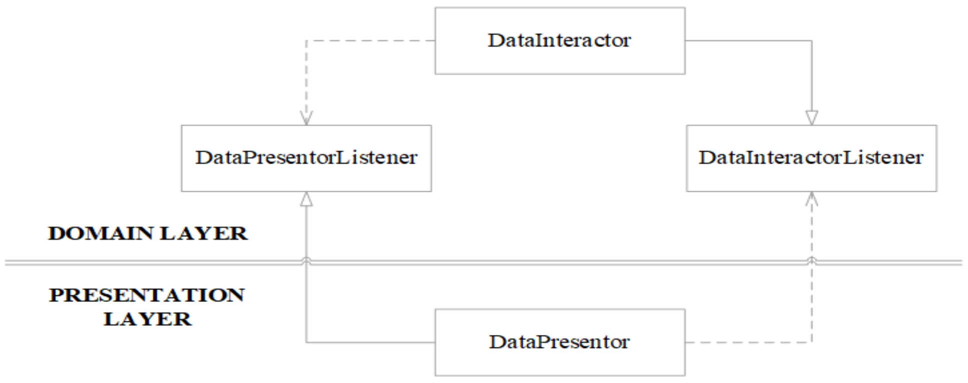

Fig. 2. The process of interaction between modules by the example of the input data unit.

Development of the Data management module
This module is responsible for the processes of loading and saving data on the hard disk of the computer. Unlike the previous modules, it is the smallest and consists of only one MainRepository class that implements the RepositoryInterface of the main module. Interaction with the Domain module takes place using superior interfaces. The process of interaction of this module with the Domain module is shown in Fig. 3.

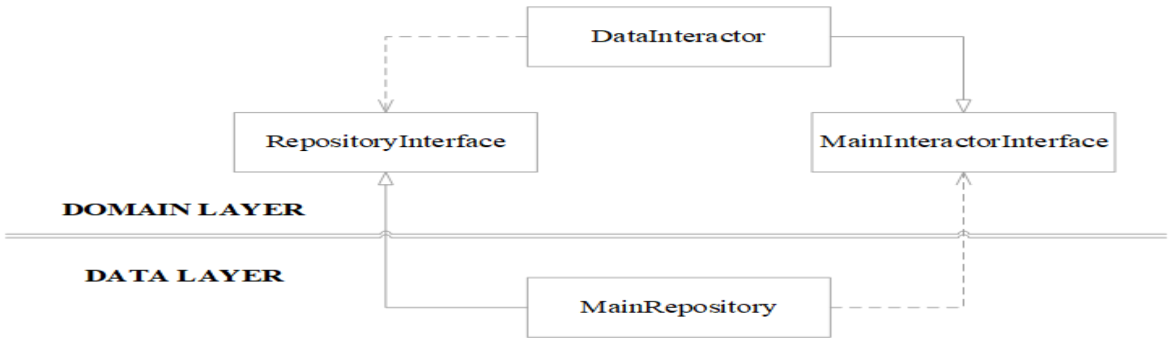

Fig. 3. Process of interaction with the Domain module.

RESULTS AND DISCUSSION
The developed graphical application combines the advantages of the considered analogues, namely:
Availability of a graphical interface;
Ability to choose the architecture of the ANN;
It does not require knowledge in the field of programming;
It supports for the Russian language.
The ability to build ANN architecture of any complexity only with the help of a graphical interface relieves its users of the need to possess knowledge in the field of programming, which cannot be offered by many existing analogues. The presence of an intuitive Russian interface simplifies the process of studying the application's tools, which will allow its users to focus on studying the principles of working with the ANN and not on exploring the application's capabilities.
None of the existing analogues of the application has the ability to observe the learning process of the ANN. However, this feature will allow its users to see the weaknesses of the developed architecture, and also to see and understand in more detail the principles of the ANN for users with minimal knowledge of machine learning.

SUMMARY
As a result of the research, an application was developed that includes tools that allow its users to create and teach ANN without having any programming skills quickly. The animated learning process makes it easier for users to choose the network architecture for the task at hand. The application works stably and without errors. To test the application, two tasks were solved: assessing the borrower's ability to fulfil their obligations to the bank, and the task of recognizing

handwritten numbers. These tasks were also solved using the Neural Network Toolbox MATLAB. ANNs were built in the MATLAB environment with the most similar structure. The results of solving the problems "Credit approval" and "Recognition of numbers" are shown in tables 2 and 3, respectively.

Table 2. Results of solving the problem "Credit approval."

Application	Working hours	Number of epochs passed	Accuracy on the learning sample,%	Accuracy on the test sample,%
Developed application	5 s	15	98.9	87
Neural Network Toolbox MATLAB	3 s	19	100	83.73

Table 3. Results of solving the problem "Recognition of numbers."

Application	Working hours	Number of epochs passed	Accuracy on the learning sample,%	Accuracy on the test sample,%
Developed application	32 min.	5	96	91.49
Neural Network Toolbox MATLAB	3 min. 25 s	10	97.66	94.95

Thus, the neural networks have learnt with the use of the developed application are not far behind inaccuracy from the networks have learnt in the MATLAB environment. However, the learning process can take much longer than its counterpart. This disadvantage can be compensated for by the fact that the developed application does not require the large computing power of the computer.

CONCLUSIONS

Thanks to the developed application architecture, further optimization of the implemented classes and the creation of entirely new ones to add missing functionality is possible.

ACKNOWLEDGEMENTS

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

REFERENCES

Asanov, AZ, Myshkina, IYu 2017, Investigation of the possibility of using neural networks in solving the problem of selecting a team for the implementation of a project. *Problems of Management*, 1: 31-39.

Ayzel, G, Heistermann, M, Sorokin, A, Nikitin, O & Lukyanova, O 2019, All convolutional neural networks for radar-based precipitation nowcasting. *Procedia Computer Science*, 150: 186-192.

Elahi, E, Weijun, C, Zhang, H & Abid, M 2019, Use of artificial neural networks to rescue agrochemical-based health hazards: A resource optimisation method for cleaner crop production. *Journal of Cleaner Production*, 238: 117900.

Goodfellow, I, Bengio, Y, Courville, A 2016, Deep learning, The MIT Press, 800 p.

Kirkpatrick, J, Pascanu, R, Rabinowitz, N, Veness, J, Desjardins, G, Rusu, AA, Milan, K, Quan, J, Ramalho, T, Grabska-Barwinska, A & Hassabis, D 2017, Overcoming catastrophic forgetting in neural networks. *Proceedings of the Russian National Academy of Sciences*, 114: 3521-3526.

Martin, R 2019, Clean architecture. A Craftsman's guide to software structure and design. St. Petersburg, Peter, Russia, 352 p.

Myshkina, IYu, Asanov, AZ, Grudtsyna, LYu 2015, Evaluation and selection of personnel based on clear and fuzzy cognitive models. *International Journal of Soft Computing*, 10: 448-453.

Nikolaeva, SG 2015, Neural networks. Implementation in Matlab: Textbook. Kazan: Kazan State Power Engineering University, Kazan, Russia, 92 p.

Rasooli, SB, Bonyad, AE, Pir Bavaghar, M 2018, Forest fire vulnerability map using remote sensing data, GIS and AHP analysis (Case study: Zarivar Lake surrounding area). *Caspian Journal of Environmental Sciences*, 16: 369-377

Rojas, R 1996, Neural Networks: A Systematic Introduction. Springer, 552 p.

Schlee, M 2018, *Qt 5.10.* Professional programming in *C* ++. St. Petersburg: BHV-Petersburg, Russia, 1072 p.

Segler, MH, Preuss, M & Waller, MP 2018, Planning chemical syntheses with deep neural networks and symbolic AI. *Nature*, 555 (7698): 604-610.

Shakla, N 2019, Machine Learning and TensorFlow. St. Petersburg: Peter, 366 p.

Stroustrup, B 2000, The C++ Programming Language: Special Edition. Addison-Wesley Professional, 1030 p.

Zhang, Q, Bai, C, Liu, Z, Yang, LT, Yu, H, Zhao, J, Yuan, HC 2020, A GPU-based residual network for medical image classification in smart medicine. *Information Sciences*, 536: 91-100.

توسعه‌ی برنامه‌ای برای ایجاد و یادگیری شبکه‌های عصبی برای کاربرد در علوم زیست‌محیطی

ایلگیز روستاموویچ سلطان بکوف*، ایرینا یوریوانا میشکینا، لاریسا یوریوونا گرودیتسینا

گروه فناوری‌های اطلاعات و سیستم‌های انرژی، موسسه نبرزین چلنی، نبرزین چلنی، ایالت تاتارستان، روسیه

(تاریخ دریافت: ۹۹/۰۲/۱۰ تاریخ پذیرش: ۹۹/۰۶/۱۶)

چکیده

روش‌های یادگیری ماشینی از رشته‌ی هوش مصنوعی نشأت گرفته‌اند و امروزه کاربرد فراوانی در چندین رشته‌ی مربوط به علوم زیست‌محیطی دارند. روش‌های یادگیری ماشینی با توجه به قابلیت مدل‌سازی غیرخطی قوی خود، امروزه در پردازش داده‌های ماهواره‌ای، مدل‌های گردش عمومی اتمسفری (GCM)، پیش‌بینی آب و هوا و اقلیم، پیش‌بینی کیفیت هوا، تحلیل و مدل‌سازی داده‌های زیست‌محیطی، پیش‌بینی هیدرولوژیک و اقیانوس شناختی، مدل‌سازی بوم شناختی و پایش برف، یخ و جنگل به کار می‌روند. در حال حاضر، محبوبیت و کاربردهای روش‌های عصبی به طور پیوسته در حال افزایش است. در این شرایط، فرایند انتخاب یک ابزار مناسب و راحت برای استفاده در علوم زیست‌محیطی با شبکه‌های عصبی، مهم است. بنابراین بیشتر ابزارهای موجود نیازمند این هستند که کاربران دارای دانش برنامه‌نویسی باشند. هیچ ابزاری برای کمک به انتخاب سریع ساختار شبکه‌ی بهینه برای مسائل حل‌شده وجود ندارد. هدف این تحقیق، ساده‌سازی فرایند انتخاب ساختار بهینه‌ی شبکه‌ی عصبی مصنوعی با توسعه‌ی برنامه‌ای با رابط کاربری گرافیکی با بازنمایی و نمایش بصری مراحل ایجاد و یادگیری شبکه‌های عصبی در علوم زیست‌محیطی است. در این مطالعه، شبکه‌های عصبی feed-foerward مصنوعی در نظر گرفته شده است. مطالعات تحقیقاتی در زمینه‌ی مطالعه، مقایسه و تحلیل ابزارهای موجود برای ایجاد، یادگیری و استفاده از شبکه‌های عصبی مصنوعی انجام‌شده‌اند. بر اساس نتایج تحقیق، یک برنامه با رابط گرافیکی باهدف حل مسائل مربوطه توسعه یافت. برنامه‌ی مربوطه بدون هر گونه مشکلی، به درستی کار کرده و امکان ایجاد و یادگیری شبکه‌های عصبی feed-foerward را بدون دانش برنامه‌نویسی می‌دهد.

*مولف مسئول

Bibliographic information of this paper for citing:

Rustamovich Sultanbekov, Il, Yurievna Myshkina, I, Yurievna Gruditsyna, L 2020, Development of an application for creation and learning of neural networks to utilize in environmental sciences, Caspian Journal of Environmental Sciences, 18: 595-601