Developing the platform model for problem solving of automated machine learning

V A Chastikova and S A Zherlitsyn
Department of Computer Technologies and Information Security, Kuban State Technological University, Krasnodar, 350072, Russia

E-mail: chastikova_va@mail.ru

Abstract. The article discusses the current state of technologies for automated machine learning. The development trends and the nature of the distribution model - MLaaS - are defined. There is highlighted a number of problems of automating the machine learning process, such as: excessive simplification and specialization of tools, vagueness of implemented processes, lack of flexibility in the infrastructure hardware, using closed algorithms. As a partial or complete solution to them, we have proposed the architecture, consisting of separate modules: models, hybridizer, learning algorithms module, testing module, user support module, and a theoretical framework. The main feature of the given architecture is its modularity, transparency and encapsulation of components. Each module is described as a separate element, implemented as an independent microservice. The paper describes the benefits of applying the given approach to the implementation of automated machine learning systems, the need to implement the given or similar standards. For each of the modules, its purposes, the tasks it solves and the implemented functionality, as well as the data necessary for the functioning and their sources are described. A general diagram showing the flows of information exchange between modules is presented. The main scenarios for the resulting system operation, as well as ways of interacting with it and the result of its operation - the generated model - are described.

1. Introduction
The range of problems solved by means and methods of machine learning is constantly growing. The given class of algorithms has repeatedly proved its efficiency not only in classical problems of clustering, classification, approximation, but also in their specific practical applications: personal identification by a photograph, by the palm vein pattern, detection of network attacks and abnormal network traffic, and many others [1].

The increased demand for such systems is confirmed by the constantly growing supply on the intelligent systems market. As an example, we can name such relatively recent services as Google's Prediction API, Microsoft's Azure Machine Learning, Amazon Machine Learning, DataRobot.com and many others.

In addition to high demand, there is also a clearly defined trend in the service distribution model in the mentioned above named tools - MLaaS (Machine Learning as a Service), which means that for the end user there are several characteristic features:

- Ease of use – to achieve success a good service should have an intuitive interface and be easy to use for its target audience, namely analysts, machine learning specialists, developers, but not
for people who are poorly familiar with data science and machine learning in particular, such as designers or managers.

- High degree of automation - most routine operations, processes and stages of development of final solutions are unified, standardized and scripted, however, often these standards remain within a particular organization.
- Cloud infrastructure - many of the discussed services offer to use their own servers to process customer data as one or even sometimes the only possible alternative.
- The closed nature of the applied methodologies and algorithms – the higher is the degree of automation, the lower is the transparency of processes, the simpler is the interface and the less obvious are processes taking place, which reduces the flexibility and productivity of services when solving non-trivial tasks, although this phenomenon is caused not only by the desire to improve the user experience, but also for reasons of preserving security or trade secrets.

As can be seen from the above list of features, each of them entails disadvantages, the effect of which could be minimized or avoided altogether.

2. Scientific merit of the issue
At the same time, there are many publications and studies aimed directly at the process of automating the machine learning process, which propose or analyze holistic solutions, such as Julia [2], AUTO-SKLEARN [3], Chameleon [4], ML-Plan [5]. Such articles propose solutions to particular problems that arise directly during the learning process, when processing or interpreting data directly, expanding the scope of application or increasing efficiency and accuracy of results by a certain fraction or number of percent.

Another category of publications on the given topic is devoted to systematizing and generalizing the already described methods for improving machine learning automation (AutoML), comparing and describing their development trends [6, 7].

However, there are very few studies aimed at systematizing and unifying approaches to the direct design of AutoML systems, architecture building patterns vary from one project to another, and the proposed solutions are often monolithic, not flexible and can be only roughly divided into blocks.

3. Problem statement
The purpose of the given work is to develop the architecture for a flexible microservice platform, consisting of isolated, extensible and replaceable modules, each of which is implemented as a separate system component and has standardized interfaces for interacting with other modules.

The proposed approach is aimed at solving the following tasks:

- Increasing flexibility of automated machine learning systems without compromising the degree of automation.
- Increasing continuity in the work of researchers and scientists by simplifying the experiment conducting.
- Bringing new studies to a standard, mutually compatible format.
- Approximating the scientific models to the form applicable in practice in accordance with the current trends in the market development.
- Minimizing or eliminating the shortcomings of existing solutions on the market in those developed according to the proposed architecture.

Flexibility of the platform is achieved due to a high degree of abstraction from the final implementation options of individual components and sets of the used learning algorithms, models and meta-learning algorithms. At the same time, due to the microservice system architecture, extensibility and replaceability of modules are provided, which makes it possible, when making improvements at one stage, not to rebuild the entire system and work only within a specific module.
Architecture standardization is an important stage in the development of the entire machine learning field and is aimed at uniting the scientific community, increasing transparency and simplifying the process of creating a reliable solution. Similar to the architectural style of component interaction of a distributed application in the Representational State Transfer (REST) network described by Roy Fielding [8], the presented architecture is aimed at providing such qualities as:

- reliability (due to the fact that there is no need to design the system from scratch and implement the existing system blocks);
- scalability;
- transparency of the interaction system (especially necessary for cloud implementations);
- simplicity of interfaces;
- portability of components;
- simplicity of making changes;
- ability to evolve, adapting to new requirements.

4. **Theoretical part**

The proposed platform includes the following set of components, hereinafter referred to as modules, regardless of the software implementation method:

- models;
- hybridizer;
- learning algorithms module;
- testing module;
- user support module (a system for choosing suitable algorithms depending on the task and input data);
- theoretical framework.

The system architecture is developed in accordance with the principles of encapsulation and responsibility partitioning. Each of the components can be represented both as a software module, an element of a monolithic program, and as a microservice. Due to the application of such approach, modularity, transparency of interaction and information flows within the system are achieved. The block diagram of the system is shown in figure 1.

![Block diagram of the automated machine learning system](image-url)

**Figure 1.** Block diagram of the automated machine learning system.
Further, each of the mentioned components is considered in more detail, also its possible content is described using the example of the author’s implementation.

**5. User support module and theoretical framework**

These two components solve the problem of increasing availability of complex intelligent models to end users who do not have expertise in the given field, without the need for in-depth study of the mathematical tools and methodology of artificial intelligence.

The user support module assumes implementation of the following interaction scheme:

- The user provides a sample of the analyzed data.
- The user answers a number of questions concerning the means of obtaining, the source and nature of the analyzed data.
- The user answers a series of questions regarding the desired final result, the nature of the expected output.
- The module generates a set of suitable models or combinations of models.
- The module generates a set of suitable learning algorithms and a set of compatibility relations of algorithms with models.
- The module forms a set of criteria for the completion of the scheme construction process.
- The module transfers the sets formed during steps 4 and 5 to the hybridization module, and it transfers those formed during step 6 to the testing module.
- The module requests help on the selected algorithms and models from the theoretical framework and provides it to the user.

![Figure 2. IDEF0 functional diagram of the user support module.](image)

**6. Model module**

The given module is a container for a variety of available implemented models. It includes the following components:

- functional models;
- probabilistic models;
- decision trees and forests;
- automatic machines;
- expert systems;
- hybrid systems.

The given list is not final and can be supplemented with other classes of models that can work in accordance with the described interface of the component interaction.
In particular, it is worth highlighting the block of neural networks, which is also included in this module. Its special feature is a variety of possible architectures, each of which can be more or less applicable to a specific task. Some of the main architectures included in the system are:

- classical perceptrons;
- deep neural networks;
- recurrent neural networks;
- convolutional neural networks;
- generative adversarial neural networks;
- Bayesian neural networks.

The contents of the model module must meet the following requirements:

The set of models has a lower limit in the form of the coverage criterion for the main set of task classes of machine learning, that is, the minimum possible set of them should be such that each of the main machine learning problems (regression recovery problem, clustering problem (unsupervised learning), identification problem, forecasting problem, knowledge extraction problem) could be solved by at least one of the models that are included in the module.

On the other hand, the more models are included in the system, the more rules must be embedded into the user support module, however, the more specialized tool will be used for solving a specific application task.

For each model there is also defined a set of hyperparameters that affect its operation. Their values are automatically adjusted in accordance with the existing limitations of the hybridization module.

Interaction with the hybridization module is carried out by means of the modification and replacement operators, which will be described below.

The IDEF0 functional diagram for this module is shown in figure 3.

Figure 3. IDEF0 Functional diagram of the model module.

7. Learning algorithms module
An important addition to the model module is the learning algorithms module, since most models cannot perform their functions directly "out of the box" and require careful adjustment of parameters, weight coefficients and other configuration values.

The main scenario for the given module operation is to perform the following set of actions:

- Getting a generated model.
- Selecting a set of compatible learning algorithms.
- Initializing the learning algorithm with the standard values of the hyperparameters.
- Starting the model training with the available data.
- Interrupting learning when low speed is detected.
- Correcting the model by means of the hybridization module.
- Correcting the learning algorithm by means of the hybridization module.
- Continuing training of the corrected system.
- Returning to step 5 or achieving the criterion of the model training completeness.

To interact with the hybridization module, small amount of data must be transmitted in both cases, the first part of which represents the current system configuration, that is, the model or optimization algorithm configuration, as well as the values of their hyperparameters. The second part contains historical data on the change in the learning progress during its execution on the configuration that was relevant at the time of exchange.

Interaction of the hybridization module with the given module is also carried out by means of the modification and replacement operators.

The given module includes the following components:

- gradient descent learning algorithms;
- genetic algorithms;
- swarm algorithms;
- hybrid algorithms;
- special algorithms.

In this case, algorithms that are designed for a specific type of model refer to special algorithms. Particular attention should be paid to the swarm learning algorithms, since due to their large number and common structure, they have a separate list of hybridization and transformation methods, which are controlled by the same-name module.

The IDEF0 functional diagram for the learning algorithms module is shown in figure 4.

![Figure 4. IDEF0 functional diagram of the learning algorithms module.](image)

8. **Hybridizer**

The hybridization module is responsible for increasing the speed, training model efficiency, and adequacy of model architecture to the task at hand. It consists of 2 parts: the first part is responsible for hybridization and modification of the model itself, its architecture and hyperparameters, and the second one is in charge of hybridization and modification of the learning algorithm in order to increase its speed and save computational resources by means of dynamic process adjustment.
The hybridization process itself is implemented via 2 higher-order operators: replacement and modification. The implementation criteria and rules for each of them are a configurable parameter and can be implemented both through an expert subsystem and through manual control.

Replacement implies a complete change of the current algorithm or model to another from a set of valid ones. In this case, hyperparameters that have mappings for this transition can be transferred to a new model, while the rest have to be initialized by means of standard values. If the given operator is implemented on the model, learning process will be reset to its initial state for a new model, since the adjusted weights and parameters have no value during a global configuration change. When implemented on a learning algorithm, the learning process can be continued from the last found position or positions.

When the modification operator is implemented, backward compatible changes will be introduced into the target component in the form of hyperparameter adjustments, minor configuration changes, or integration of an additional submodel or an auxiliary learning algorithm.

As it has been already noted above, the block of swarm intelligence algorithms plays a special part in interaction with this module. It is distinguished by a greater flexibility of modification and integration methods, algorithms of the given type among themselves. Such feature is due to the uniform implementation structure of most swarm systems, as well as a large number of strategies for their combination. For example, it is possible to replace some of the search agents with agents of a different type, which will carry out their movements across the landscape of the optimized model in accordance with their own algorithm, but not with the initial one.

The IDEF0 functional diagram for the hybridization module is shown in figure 5.

![Figure 5. IDEF0 functional diagram of the hybridization module.](image)

9. Testing module

The task solved by the given module is to store and check the quality criteria fulfillment of the developed model. After passing through each stage of the system operation, the output is a ready-made model, which must be checked for adequacy according to a number of criteria generated on the basis of the input data. At this stage, the results of work on the validation subsample are assessed, and the ratio of the number of errors of different types is checked.

The given module is also used for early completion of the learning procedure, as a tool to reduce the time spent and computing resources. To do this, once in a given number of learning cycles, the model is fixed and completely transferred for analysis to the described module. The module, in its turn, performs tests to check the existing quality criteria and concludes on necessity to continue model training.
Similar scenario is implemented during the final check, however, if a number of criteria are not met, the model returns to the learning stage, and probabilities of choosing the components that led to the given result are reduced.

In addition to the described functionality, the given module implements the user interaction interface with the model after completing its training, as well as integration with external services.

The IDEF0 functional diagram for the testing module is shown in figure 6.

![IDEF0 functional diagram of the testing module](image)

**Figure 6.** IDEF0 functional diagram of the testing module.

### 10. General scheme of module interaction

The system architecture is deliberately divided into separate modules, which, when interacting, allow in an automated mode solving a number of applied tasks that require the construction of intelligent models. The functionality of each of the blocks has been described above. Figure 7 shows a scheme of their interaction and the flows of transmitted information.

![Scheme of interaction of the system components](image)

**Figure 7.** Scheme of interaction of the system components.
In the presented image, the functional blocks are colored in blue, the input and output data – in orange. White blocks describe the information transmitted between the components and the intermediate results of their work.

The output and final result of the system operation is always some information model, built and trained on the example of input data and ready for operation. It can be used both by means of the testing module through the graphical interface or API that it provides, and by means of import and implementation into another software product or service.

11. Conclusion
In the given paper a platform model for solving automated machine learning problems is described. There has been proposed a set of architectural and functional components that, when integrated, form a flexible, scalable, universal infrastructure that provides encapsulation of the constituent systems. The finite sets of possible module interactions between themselves, as well as the principles of dividing the functionality between them have been described.

References
[1] Chastikova V A, Zherlitsyn S A and Volya Y I 2018 Neural network method of identification by unformalized semantic characteristics News of Volgograd State Technical University 8(218) 63-7
[2] Rackauckas C, Anantharaman R, Edelman A, Gowda S, Gwódz M, Jain A, Laughman C, Yingbo Ma, Martinuzzi F, Pal A, Rajput U, Saba E and Shah V B 2021 Composing Modeling and Simulation with Machine Learning in Julia ArXiv abs/2105.05946
[3] Feurer M, Klein A, Eggensperger K, Springenberg J T, Blum M and Hutter F 2015 Efficient and Robust Automated Machine Learning NIPS
[4] Otterbach J and Wollmann T 2021 Chameleon: A Semi-AutoML framework targeting quick and scalable development and deployment of production-ready ML systems for SMEs ArXiv abs/2105.03669
[5] Mohr F, Wever M and Hüllermeier E 2018 ML-Plan: Automated machine learning via hierarchical planning Machine Learning 107 1495-515
[6] Bie T D, Raedt L D, Hernandez-Orallo J, Hoos H, Smyth P and Williams C K I 2021 Automating Data Science: Prospects and Challenges ArXiv abs/2105.05699
[7] Xin D, Wu E Y, Lee D, Salehi N and Parameswaran A G 2021 Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows Proceedings of the 2021 CHI Conference on Human Factors in Computing
[8] Fielding R T and Taylor R N 2000 Principled design of the modern Web architecture Proceedings of the 2000 International Conference on Software Engineering (Ireland: Limerick) pp 407-16