# Principles of decision-making systems building based on the integration of neural networks and semantic models

Vladimir Golovko
Alexsander Kroshchanka
*Brest State Technical University*
Brest, Belarus
gva@bstu.by
kroschenko@gmail.com

Valerian Ivashenko
Mikhail Kovalev
*Belarusian State University of Informatics and Radioelectronics*
Minsk, Belarus
ivashenko@bsuir.by
michail.kovalev7@gmail.com

Valery Taberko
Dzmitry Ivaniuk
*JSC ≪Savushkin product≫*
Brest, Belarus
id@pda.savushkin.by

***Abstract*—This article reviews the benefits of integration neural network and semantic models for building decision-making systems. There purposed an approach to the integration of artificial neural networks with knowledge bases by inputs and outputs and the specification of these networks in the knowledge base using the ontology of the respective subject domains. The proposed approach is considered on the real production problem of JSC ≪Savushkin Product≫ for quality control of marking on products.**

***Keywords*—ANN, knowledge base, integration, inference, decision-making**

## INTRODUCTION

Despite the significant results obtained in different directions of research in the field of artificial intelligence, the problem of integration of such results is gaining more and more relevance. A large number of problems that should be solved by modern intelligent systems (IS) require the joint use of different models of problem solving and models of knowledge representation. In turn, the integration of different models of this kind within a single system often presents significant challenges due to the development's isolation of these models.

Currently, one of the most actively developed directions in the field of artificial intelligence is the direction related to problem solving based on machine learning methods. The popularity of methods for problem solving based on machine learning is largely due to the development of theoretical models of artificial neural networks (ANNs) and productive hardware platforms for their implementation. The variety of architectures, methods, directions and ways of ANNs using is constantly increasing.

However, it should be noted that not all problems are convenient to solve with the using of machine learning because the complexity of modern tasks creates the need to integrate different approaches to problem solving. It is the common situation when systems, that use neural network algorithms, are in need for additional semantic analysis of results of ANNs work and decision-making on the basis of this analysis.

In this regard, there is a need to develop approaches to the building of systems that can use both neural network and semantic models, as well as able to combine these models in the search for problem solving. There are two main requirements for such a system:

- flexibility in adding new models;
- adaptivity to existing models changing.

This article will consider an approach to the building of such systems on the example of a subsystem of marking quality control for the company JSC "Savushkin product".

## I. FORMULATION OF THE PROBLEM

### A. General statement of the problem

This article discusses the use of integration of artificial neural networks with knowledge bases to solve problems of a particular class, the general condition of which can be formulated as follows: it is necessary to perform a semantic analysis of the results of the machine vision system with use of the knowledge available in the system.

In the formal way the general condition can be formulated as shown below. There is a signal of fixed size $s$. For the machine vision system it is required to find the transformation $f$ of this signal satisfying the set of constraints $R_f$ in the feature space [1]:

$F^{q*d}$

$F \subseteq S \times V$

where $q$ is the number of simultaneously tested signals, $d$ is the length of the time interval, $S$ is the set of signals, $F$ is the set of product features, $V$ is the set of feature values. Moreover, it is required that $f$ should be such that there is a transformation $g$:

$g \subseteq V^{q*d} \times D$,

satisfying a set of constraints $R_g$, where $D$ is a set of decisions, the simplest of which are:

- notifying the operator;
- stopping the process;
- making fixes in some devices if possible;
- moving the video camera;
- cleaning the video camera;

### *B. Approach to solving the problem*

This problem solution consists of two parts, described by transformations $f$ and $g$, the implementation of which must be integrated to obtain a system that provides a complete decision. In this paper, it is proposed to use models based on artificial neural networks to implement the transformation $f$, respectively, to implement the transformation $g$, it is proposed to use ontologies based decision-making models. Thus, the results obtained by the authors in [1], devoted to the integration of ANNs with knowledge bases, will be used in this work.

This integration can be considered at different stages of building partial decisions:

- the training of ANNs for recognition;
- data supply to the input of ANNs;
- ANNs work results processing
- decision development on action or inaction on the basis of the knowledge base.

The first step is to select and train the model to solve the first part of the problem. Integration is appropriate if these choices and training are partially automated. For example, under the condition that there is a system with the knowledge base, which knows what models (ANNs) can be used to solve the formulated problem and compare on the grounds that there are restrictions in $R_f$. Or, the system knows which training methods can be applied and for which data sets (training samples), in which case the system can manage the training process by passing this data for training and testing.

The second stage is also reduced to ANNs management and integration is appropriate in the presence of a system with the knowledge base, storing knowledge about the input data (eg, video signal).

At the stage of processing the results of ANNs, if the network solves the problem approximately, and not exactly, there may be situations when the results are not validated on the data that were not included in the training (or test) sample. In this case, integration is possible if the knowledge base contains knowledge about the required feature properties from the set of $R_f$ constraints. If invalid conditions are detected, the network can be adapted.

At the last stage, the feasibility and complexity of integration depend on the type of model used for the solution. In this case, the options for selecting the model are as follows:

- ANNs similar to ANNs of the first subtask;
- ANNs is other than ANNs of the first subtask;
- not adaptable (trainable) model that are not related to ANNs;
- adaptable (trainable) model that is not related to (being) ANNs.

The first option does not require serious integration or requires integration by inputs and outputs, the second option requires integration by inputs and outputs, the third and fourth options require the most serious work on the integration of models. The fourth option, unlike the third one, does require the use of flexible knowledge-based systems with developed means of knowledge representation and adaptation of the knowledge base. The complexity of integration in this case will depend on the choice of a specific model to solve the second sub-task (decision-making). These models can be attributed to:

- problem solving strategies as decision trees and others;
- classical and non-classical inference models, including fuzzy and Bayesian models of reasoning, non-monotonic reasoning systems etc.;
- and others.

## II. Existing approaches to solving the problem

### *A. Integration approaches*

The main approaches to integration are the following:

- integration by inputs and outputs, including control and messaging transfer;
- integration by immersion, embedding one model into another when one model is modeled (interpreted) by another.

Details of the implementation of these approaches depend on the complexity of selected integrated models: language, set of states and set of operations. For complex models, immersion is a labor-intensive process, so the choice of integration by inputs and outputs can be due to reduced labor costs.

The advantages of the approach of integration by inputs and outputs are the possibility of reducing the labor costs of integration and higher performance of the decision making.

### *B. Problems encountered in the task solving*

The main problems that are solved by the models integration:

- the reduction of labor costs for the complete decision making model development;
- satisfying restrictions on transformations describing decisions, including its computational complexity, in order to improve the efficiency and performance of the entire system;
- obtaining a decision that has a model, i.e. the decision implementation is not just a computational

process, but a formula that has a model semantics in a certain model of knowledge representation with the necessary properties, in order to conduct a semantic analysis of the obtained decision and explain its results and continue the process of adaptation and integration.

The benefits are reduced labor costs, finding the most efficient (productive) solution, and building a flexible system focused on these benefits.

## III. Proposed approach

### A. Detailed problem description

The task of marking control is described in detail in [1]. Brief description – the marking control module is installed on the bottling line. It consists of a camera connected to an industrial computer. The camera is fixed in the box and installed above marked covers. The system shall ensure continuous monitoring of marking quality. Next, we consider in more detail the analysis of the main reasons for obtaining defective marking and typical ways to eliminate them.

- **no ink.** If empty bottles start to go, it means the printer is out of paint. The system can access it additionally (since the printer is connected to the network) and check the availability of ink.
- **camera shift.** the system knows that batch bottling has started, but there are no positive recognition results from the camera.
- **incorrect marking.** The marking is recognized, transferred to the system, but it does not match the reference – so there was an error when specifying the text marking and it is necessary to stop the bottling process and notify the operator.
- **unreadable marking.** The marking is not recognized – one or more digits are not recognized, so the printer nozzles are clogged – you need to stop the filling process and notify the Instrumentation engineer about the need to clean the printer nozzles. It is desirable to remove the bottles with the incorrect marking of the pipeline.

### B. Technology

The proposed approach is based on the OSTIS technology ( [2], [3], [4]) and its principles. The OSTIS technology uses models of knowledge representation and processing focused on the unification and work with knowledge at the semantic level. The basic principles and models used in the approach include:

- knowledge integration model and unified semantic knowledge representation model [1], which is based on the SC-code [2];
- principles of situational management theory [5];
- ontological model of events and facts in knowledge processing [6];
- multi-agent approach [7];
- hybrid knowledge processing models [8].

## IV. Architecture of the proposed system

As mentioned above, the proposed subsystem of marking quality control is developed on OSTIS technology.

The system, developed on OSTIS technology, is called ostis-system. Each ostis-system consists of a platform-independent implementation of a unified logical-semantic model of the system (sc-model of a computer system) and a platform for the interpretation of such models. In turn, each sc-model of a computer system can be decomposed into sc-model of knowledge base, sc-model of problem solver, sc-model of interface and abstract sc-memory which stores the constructions of the SC-code [2].

Based on this, the developed system, like any ostis-system, has three main parts:

- **sc-model of interface.** Describes the SCADA-system project [9], which can be used to track the decisions made by the system, including decisions about the need of an engineer's involving. Also here the engineer can set a sample to configure the system to recognize markings on new products.
- **sc-model of knowledge base.** Describes the knowledge base [10] of the system, which contains all the necessary knowledge for decision making, such as logical rules, statements, current markings, device states and etc. The knowledge base of the system of quality control marking is described in the VI section.
- **sc-model of problem solver.** Describes problem solver based on the multi-agent approach ( [8], [11]). Contains a set of internal and external agents that work with the knowledge base. With the help of these agents, problem solver can initiate the recognition process, implement the reverse inference ( [12], [13]) on the recognition results, call external programs for decisions implementation, prepare a decision for the engineer's terminal. The problem solver of the subsystem of quality control marking is considered in more detail in the VII section.

Also system has additional modules such as:

- device serving image marking. For the knowledge base, these devices are represented by agents that are initiated by the recognition request.
- marking recognition module. The task of this module is to localize and recognize product markings on the image. This module is described in more detail in V section.
- robotic subsystem control module. This module has access to the subsystem that directly carries out the marking of products. The task of this module is implementation of system decisions that can be taken without the involvement of the engineer, such as marking updates, switching the subsystem on and off, etc.

The figure 1 shows the general diagram of all modules interaction of the system of marking quality control.

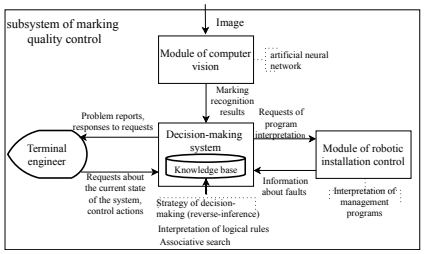


Figure 1. Diagram of all modules interaction of the system of quality control marking

## V. Structure of the recognition module

To solve the problem of caps marking detection we use two different approaches. One of them is a component of the real system, which already used in the work. Another approach is still being explored and this section is dedicated to it.

To solve caps detection task we used deep neural network SSD (Single-shot detector) [14]. A main feature of this architecture is that it detects objects in images in one pass (one-look), without solving two independent tasks (localization and classification). Using of such network gets acceptable speed of objects detection, and with modified classificator (for example, Tiny SSD or MobileNet SSD) allows to work in real-time mode ( [15], [16]). Use of networks such as Faster-RCNN [17] gives better efficiency, but fundamentally unacceptable for real-time applications because of the high resource intensity [18].

SSD-network as the YOLO-network [19], belongs to category of one-look methods, which solve detection task with using only one network. A schematic representation of the architecture is shown in figure 2. SSD has a typical structure inherent to convolutional neural networks [20].

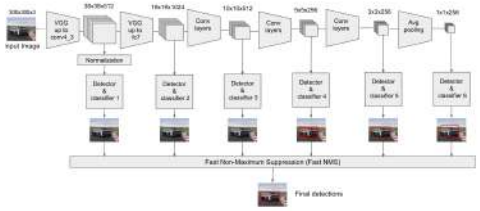

Figure 2. SSD-network architecture

On output of SSD-network we gets coordinates of rectangle boxes, which contains objects and labels for each box, which represent class of object.

We note main features of this network.

- It differs from other single shot detectors (in particular, from YOLO) in that each layer of the model participates in the formation of information about objects and their location (taking into account the scale of these objects – each subsequent layer detects objects of larger size than the previous one) (figure 3);
- It uses pretrained neural network (VGG16 [21], ResNet50 [22], etc.) as a base item, which is integrated into the SSD network and forms features maps that are used to make decisions about the position and class of objects. These networks trained on massive set of images. As a alternative path to get pretrained network is a use of special pretrain procedure [23]. Layers for classification are discarded;
- The network uses the Non-Maximum Suppression algorithm to reduce the number of generated boxes;
- Each item of feature map forms set of default boxes (or anchors), which differ in scale and aspect ratio.
- Network is trained until for each anchor gets right prediction for its class and offset.

Although SSD-network with VGG16 works faster in contrast with networks, which use a two-stage process, this architecture cannot be considered as a working version to detect in a simple mobile systems. In this case it seems appropriate to use the MobileNet as a basic convolution network. MobileNet can significantly speed up SSD-network, therefore makes possible to process the video stream received from the camera over the pipeline in real-time mode. The main feature, due to which an increase in processing speed is achieved, is that MobileNet contains fewer number of parameters compared to VGG16, but it is not inferior to it in efficiency [16]. Reducing the number of parameters in turn is achieved by using a Depthwise Separable Convolution.

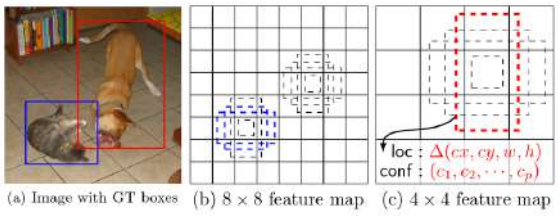


Figure 3. Localization of objects on feature maps of different sizes [14]

*Structure of the caps detection system.* SSD-network has one significant drawback – poor ability to detect small objects in the image ( [24], [25]) This is due to the fact that the feature maps of this network have low resolution. Therefore, when designing a system for the detection of caps and labels with the subsequent recognition of the individual chars, which are included in the label, we must perform the decomposition of the general task into two subtasks.

- Detection of caps and labels. At this stage the

missing labels are detected and a decision is made to send message to the operator about incorrect cap.
- Detection and recognition separate chars. The detection and recognition of chars and the formation of the marking representation are perform at this stage.

Both of these tasks can be solved using SSD-network. But in the case of using one detector for detection and recognition of all objects of interest (caps, labels, individual characters) is impossible to ensure acceptable quality of work for individual characters, because the original image will be scaled to the size, which makes the detection of the symbols a difficult task.

Therefore we used two separate SSD-networks for detection of caps/labels and recognition chars. First network detects two classes of objects (caps and labels), then feeds image of label to second network in original quality.

*Description of training set and features of labeling* To solve the task of detection of caps and labels, the training set preparation consisted in manual processing of images with the definition for each image characteristics of rectangular areas (boxes), which include caps and labels (length, width, coordinates of the upper left corner).

As a result, a total set of 940 images was prepared, 80% of which form a training set, and the remaining 20% - a test set.

Example of image labeling is presented in fig. 4.

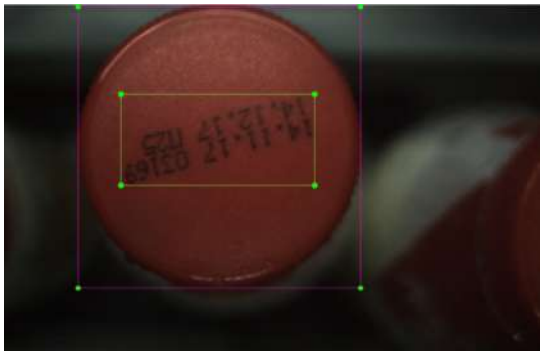

Figure 4. Example of image labeling

*Estimation of recognition efficiency* The $mAP$ (mean average precision metric) was used to evaluate the detection quality of caps and labels. This metric is the de facto standard of metrics used to evaluate the quality of models used for detection [26]. It is used together with its modifications computed for various threshold values of $IoU$ (Intersection over Union, a quantity called the Jaccard index). Value of $IoU$ is calculated by the formula:

$$IoU = \frac{S_{ground\_true} \cap S_{box}}{S_{ground\_true} \cup S_{box}} \quad (1)$$

where $S_{ground\_true}$ defines the area of the etalon box, which uses to labeling of the training set, and $S_{box}$ is the area of the detected box.

As you know, the precision is calculated by the formula

$$P = \frac{TP}{TP + FP} \quad (2)$$

where $TP$ and $FP$ denote, respectively, the number of true-positive and false-positive detection results, and, accordingly, $P$ determines the share of correct detections in the total number of detections received by the neural network.

With respect to the object detection task, the number $TP$ determines the total number of rectangular areas for which the value of $IoU$ calculated with respect to ground-true boxes is greater than some given threshold (usually the threshold of 0.5 is chosen). Thus, if the value of $IoU$ for such a predicted region exceeds 0.5, the detection is considered to be true-positive. If there are several detections for a given true region, then one detection with the largest value $IoU$ is selected, and the rest are considered as $FP$.

The averaged value for all values of recall gives the $AP$:

$$AP = \frac{1}{N} \sum_{i=1}^{N} \frac{TP_i}{TP_i + FP_i} \quad (3)$$

where $N$ is the number of equally spaced recall values.

The value of $mAP$ is obtained from $AP$ by subsequent averaging over all available classes (for the solved problem of such classes two are objects ”cover” and ”label”).

*Results of objects detection and recognition* After training the SSD-network for detection of caps and labels we have got the results of detection with efficiency in mAP = 0.98 (on both classes). This is the result, which in most cases guarantees an acceptable quality of detection with the possibility of application in real systems. Examples of detection are presented in Fig. 5.

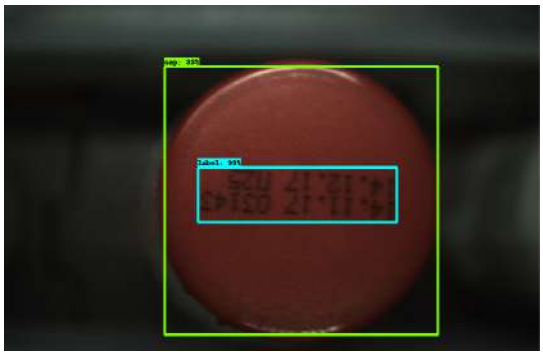

Figure 5. Result of the detection objects by using SSD-network

SSD-network was trained during 4000 iterations, on each of which the adjustment of network parameters was

made for mini-batch, with size of 8 images. Evolution of the mAP presents in fig. 6.

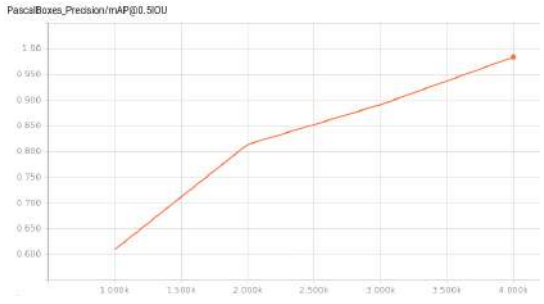


Figure 6. Evolution of the mAP during training

## VI. Structure of the decision-making system knowledge base

### A. Representation of the trained ANN in the knowledge base

At this stage of integration, the knowledge base stores the trained ANNs and following data about its:

- type of ANN;
- type of input data;
- set of recognized classes or output type;
- additional identifiers for linking objects in the knowledge base and corresponding recognition engine components

At this stage, it is necessary to specify the type of ANN when there are several different ANN of a certain type in the system and there is a need to make a decision on the use of a particular network. The system should independently make a decision on the basis of knowledge about the recognized object, and therefore there is a need to describe in the knowledge base the entire hierarchy of subject domains of ANN proposed in [1]. With the help of this hierarchy of subject domains, it is possible to describe such important for making decisions knowledge about the use of the trained ANN as:

- class of problems solved by ANN;
- architecture of ANN;
- the average work time of ANN;
- quality of recognition;
- and others.

In addition, such a detailed description of the trained ANN in the knowledge base can be used to provide information support to the engineer who will update the architecture or retrain the ANN.

A lot of IDs in an external module is used to communicate the results of the recognition module with fragments the knowledge base. For example, it is necessary to make a clear matching between the recognized classes in the recognition module and the corresponding classes in the knowledge base. Different recognizable classes from different trained ANN in the recognition module can correspond to the same recognized class from the knowledge base. Naturally, when moving to the next stage of integration of semantic and neural network models and the implementation of neural network models on OSTIS technology, the need for a set of integration identifiers will disappear, and the system is designed in advance so that the removal of this set occurs with minimal effort.

Figure 7 shows a piece of the knowledge base that describes two trained ANNs, where one classifies the image and the other classifies the text. Also, each network has a set of recognizable classes that intersect in the knowledge base, but these classes are different in the recognition module.

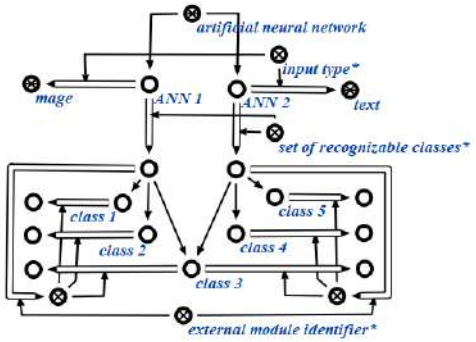


Figure 7. Representation of trained ANNs in the knowledge base

### B. Representation of logical rules for decision making

To make decisions, the system must have a set of implicative and equivalence tuples, which for brevity will be called logical rules. According to these rules, the reverse inference is used to make the decision or a set of decisions. The rules are described in the knowledge base using constant and variable sc-nodes and sc-connectors. Inference works on the basis of such rules. Inference uses the ≪if≫ part as search patterns in the knowledge base. When matching the ≪if≫ part of a statement was found, the system generates the knowledge described in the ≪then≫ part of the implicative bundle of the used logical rule. For logical rules, presented in the form of equivalence tuples, the mechanism of its using is similar, with the only difference that in place of the ≪if-then≫ parts there can be any part of the equivalence tuple.

It should be noted that the logical rule can be the specification of agents or programs. These specifications are represented in the form of the implicative tuple, in which ≪if≫ part describes the input data, and in ≪then≫ part describes the output data. When making the inference, the problem solver will use these logical rules on a par with the rest, but when using these logical rules, the appropriate agent or program will be called.

Each logical rule has the number of times it is used by inference. This technique will allow the system to self-learn and speed up the inference for trivial situations.

Logical rules are related to the specific subject domain in which its are used. In the case of the task under consideration, this is the subject domain of product marking. Figure 8 shows a fragment from this subject domain that describes the set of recognizable **yogurt bottles** of a certain class, its current standard marking of the current product that will be tested, the marking photo and the marking recognition result by module's recognized.

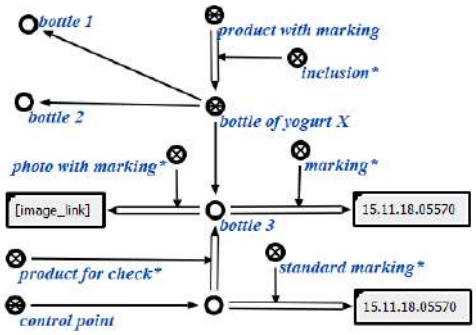


Figure 8. Fragment of the subject domains of product marking for bottles of yogurt **X**

The knowledge presented in figure 8 creates a pre-requisite for the use of a logical rule for checking of marking compliance to standard(figure 9). This rule is: *matching the marking of a product from a certain subset of the marked product with the standard marking of this subset is equivalent to the fact that this product belongs to a set of correctly marked products.* This rule should be used after the recognition is completed in the first place to effectively handle most of the situations that arise in the system when the products are marked correctly.

However, before using the rule shown in figure 9, it is necessary to compare the recognized marking with the standard, for which it is needed to use a simple program of string comparison. This program is bound to a logical rule (figure 10), which says that if you apply to the input of this program two strings, it will generate in memory one of the three structures:

- strings are equal;
- the first string is greater than the second in lexicographical order;
- the first string is smaller than the second in lexicographical order.

### C. Representation of the system work result

One of the most important features of the system is the ability to explain made or proposed decisions. For this purpose, the inference makes a decision tree in the course. Decision tree stores the input recognition data,

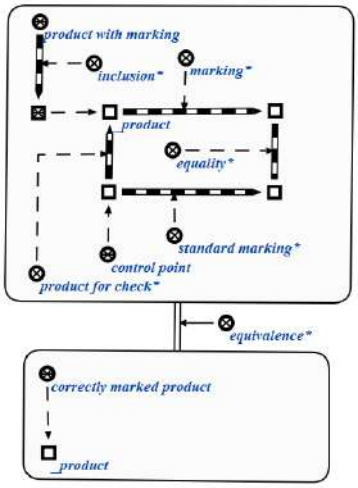


Figure 9. Logical rule for checking of marking compliance to standard

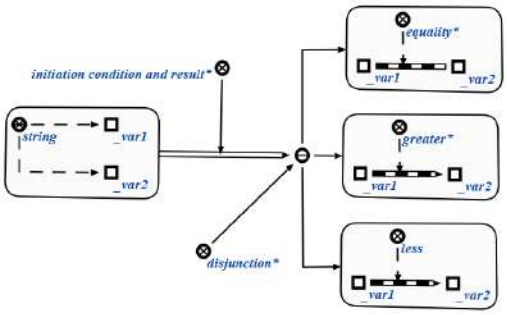


Figure 10. Logical rule of using a program compared to two strings

the recognized marking, the chain of applied logical rules and the applied (proposed for application by the engineer) decision.

With the help of the decision tree, it becomes possible to restore the chain of logical rules that led to the final decision. The restoration of such a chain is not a trivial task, since the use of reverse inference may lead to deadlocks after using of logical rules, and the knowledge generated as a consequence of using these rules may be a prerequisite for using of other logical rules which will lead to the final decision (or decisions).

## VII. Problem solver of decision-making systems

The general problems of the decision-making system's problem solver are:

- access to the knowledge base;
- processing (analysis, verification and generation) of knowledge;
- interaction with other modules of the system.

In OSTIS technology, problem solvers are constructed on the basis of the multi-agent approach. According to this approach, the problem solver is implemented as a set

of agents called *sc-agents*. All sc-agents interact through common memory, passing data to each other as semantic network structures (sc-texts).

Initiation condition of the sc-agent is some event in the knowledge base. These events are changes of temporary non-belonging to the temporary belonging of the element to the situative set, which is interpreted as a set of sc-agent initiation commands. Each command represents the data that will be processed by the sc-agent. This data can be a single sc-element and its semantic neighborhood available in common memory, or some structure (sc-structure) linked by such a sc-element [8].

After the operation started and executed, the temporary belonging is replaced with a temporary non-belonging, however, new temporary belonging may appear that will initiate the work of other sc-agents.

Sc-agents can be divided into external and internal. External sc-agents interact with the knowledge base but are its not part of it. Internal sc-agents are part of the knowledge base and can be implemented using the internal language called SCP.

It should also be noted that some agents may be non-atomic. This means that two or more other sc-agents are used to implement its functionality.

Copies of the same sc-agent or functionally equivalent sc-agents can operate in different ostis-systems, while being physically different sc-agents, since it is assumed that the proposed system can be used in other systems of quality control marking, and some sc-agents can be used in other decision-making systems. Therefore, it is advisable to consider the properties and typology not of sc-agents, but of classes of functionally equivalent sc-agents, which we will call *abstract sc-agents*.

### *A. Abstract non-atomic sc-agent of marking quality control system*

The whole problem solver of the system under consideration can be represented as a decomposition of an abstract non-atomic sc-agent of quality control marking system, which is presented in figure 11.

Before going to the consideration of the main sc-agent, which is an abstract non-atomic sc-agent of quality control marking system, there is a brief description of the rest:

*1) Abstract sc-agent of initiation the marking verification process:* external agent who receives a photo of the marked product from the camera and creates a semantic structure in the knowledge base that describes the product for verification, a photo with its marking, as well as the source of the photo (number of the camera from which the photo was taken).

The figure 8 presents an example of sc-structures, which this agent creates in the memory.

*2) Abstract sc-agent of the search ANN for recognition:* internal sc-agent that reacts to the appearance of a new product with a photo of marking. Since different

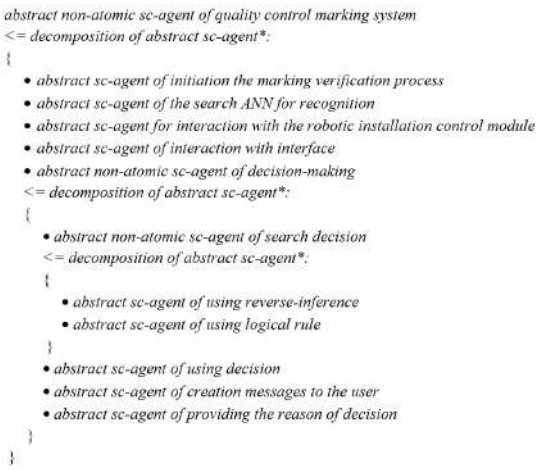


Figure 11. Decomposition of abstract non-atomic sc-agent of quality control marking system

ANNs(different products, cameras and shooting angles) can be used to recognize different marking sources, the agent finds a suitable ANN in the knowledge base and initiates its using with the help of an abstract sc-agent interaction with the recognition module.

*3) Abstract sc-agent of interaction with the recognition module:* external sc-agent that reacts to the events of ANNs usage from the recognition module. Receives the pointer on the necessary ANN and input data. After finish ANN work, it put the result of its work into the knowledge base and then initiates the work of an abstract non-atomic sc-agent of decision-making.

*4) Abstract sc-agent for interaction with the robotic installation control module:* external sc-agent, whose task is to make various external requests to the robotic installation control module, such as checking the level of paint in the printer, changing the marking label or rejection of a certain bottle.

*5) Abstract sc-agent of interaction with interface:* internal sc-agent, whose task is to implement user requests to the knowledge base.

### *B. Abstract non-atomic sc-agent of decision-making*

The main task of the abstract non-atomic sc-agent of decision-making is to develop and apply a decision about the quality of the marking applied to the products. In turn, this agent is decomposed into:

- Abstract non-atomic sc-agent of search decision;
- Abstract sc-agent of using decision;
- Abstract sc-agent of creation messages to the user;
- Abstract sc-agent of providing the reason of decision.

Abstract non-atomic sc-agent of search decision is work on development of decision and compilation of decision tree. Its functionality is implemented with two agents:

*1) Abstract sc-agent of using reverse-inference:* the work of this sc-agent is initiated by an abstract sc-agent of interaction with the recognition module immediately after adding the recognition result to the knowledge base. Figure 8 shows an example of the initial knowledge with which this agent starts working.

The search for a decision is divided into two stages. At the first stage, sc-agent checks whether the products are correctly marked. At the second stage, which begins only if the product is not correctly marked, there is a search for the decision to the situation.

Since the sc-agent uses reverse inference, it starts with the final states. At the first stage, it tries to use logical rules before appearing one of the following semantic constructions in the knowledge base:

- the product belongs to set of correctly marked products(figure 12);
- the product does not belong to set of correctly marked products(figure 13).

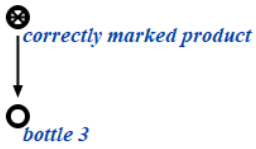


Figure 12. Example of the semantic structures describing the belonging of the product to set of correctly marked products

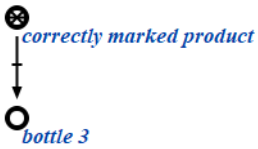


Figure 13. Example of the semantic structures describing not belonging of the product to set of correctly marked products

At the start work of agent tries to create sets of logical rules(by pattern search), the application of which will lead to the appearance of the necessary semantic construction in the knowledge base. Next, it tries to apply the most frequently used logical rule. The logical rule can be applied when there is semantic construction in the knowledge base that isomorphic to the construction that was obtained by substituting nodes associated with the processed product into a template from a logical rule. This pattern is the first part of the logical rule, the second part describes the knowledge that will be generated after applying this logical rule.

If the rule can be applied, the system initiates abstract sc-agent of using logical rule, and adds the logical rule and the result of its using to the decision tree.

In the case when there is not enough knowledge to apply a logical rule in the system, the agent recursively initiates the work of itself, where it is already trying to find logical rules, the application of which will lead to the appearance of the missing knowledge in the knowledge base.

If the using of any logical rule does not result in the appearance of the necessary semantic constructions in the knowledge base, the agent reports that it can't find the decision for this problem.

In the second stage of search system uses the same principle, only the recursive search of logical rules starts with another set of final states:

- send messages to the engineer about the lack of paint in the printer;
- send a message to the engineer about shifting camera;
- request permission from the engineer to update the printed marking;
- request permission from the engineer to update the standard marking;
- ignore.

This list can be expanded after adding new logical rules to the system.

*2) Abstract sc-agent of using logical rule:* sc-agent receives the logical rule and the matching constant nodes to variable nodes of one of the logical rules parts. Next, it uses a logical rule, which is expressed in one of three actions:

- if the logical rule is implicative or equivalence bundle, the sc-agent will substitute the matched constant nodes into the logical rule nodes and generate the knowledge described in the "to" part(for the equivalence bundle, the if-then roles are specified at the sc-agent input for each part of the bundle);
- if the logical rule assumes the call of any program, it is called with the matched input data, after which the sc-agent waits for program completion;
- if the logical rule assumes the initiation of any sc-agent, it is initiated with the matched input data, after which the main sc-agent waits for the completion of the initiated sc-agent.

After applying the logical rules, the agent increases the use count of this logical rules and exits.

Next, let's take a brief look at the work of other sc-agents:

*3) Abstract sc-agent of using decision:* the task of this sc-agent is using of the developed decision in the process of inference. The decision could be sending a message to the engineer, reaction to the response of the engineer, the automatic rejection of products, the change of a marking label, etc.

*4) Abstract sc-agent of creation messages to the user:* this sc-agent is initiated by the abstract sc-agent of using decision in the case when there is a need to compose a message to the user whose template was defined at the stage of decision search.

The task of the sc-agent is to substitute the necessary variables into the message template. Variables can be: control point, printer, paint level in the printer, etc.

*5) Abstract sc-agent of providing the reason of decision:* sc-agent is initiated by user request when he wants to see the reasons for decision-making. Sc-agent receives the node of the decision, from which the decision tree is easily extracted.

The task of the sc-agent is to transform the decision tree into a view that is readable for user. To do this, the sc-agent creates a chain of logical rules from the decision tree, the using of which led to the decision and cuts off the logical rules, the using of which did not help in making the decision. Sc-agent made naturally language text for on the basis of natural language formulations that each logical rule has.

## VIII. Example of system's work

Let us consider an example of the system operation on the case of the camera shift on the tape for marking of some yogurt bottles. The engineers set up the system to receive a message with a marking problem if problems were found with the recognition of three consecutive bottles.

The system gets to check the next bottle. System create a description in the knowledge base that includes:

- photo of the bottle cap with lint to the camera from which the photo was taken;
- specify the type of product that the bottle is;
- specify the control point where the photo was taken.

After that, the photo of the bottle is transferred to the recognition module to determine the bottle marking correctness.

Figure 14 shows a part of the knowledge base that describes the knowledge about the processed bottle after finish work of the recognition module, which shows that the module was unable to recognize the marking.

After finish work of the recognition module, the problem solver starts searching for a decision using the reverse-inference, the logic of which is described in the VII section.

First of all, the problem solver tries to use all logical rules in which the concept of *correctly marked products* is involved. The solver tries to apply the rule of checking for compliance with the standard marking, shown in figure 9.

Since it is not known whether the recognized marking is equal to the standard one, solver call the lexicographic string comparison program, the specification of inputs and outputs of which are presented in figure 10.

Because the strings are not equal, the solver tries to use the following logical rule: *if a product belongs to a set of consecutive unmarked products at its control point and the capacity of this set is greater than the allowable number of consecutive unmarked products,*

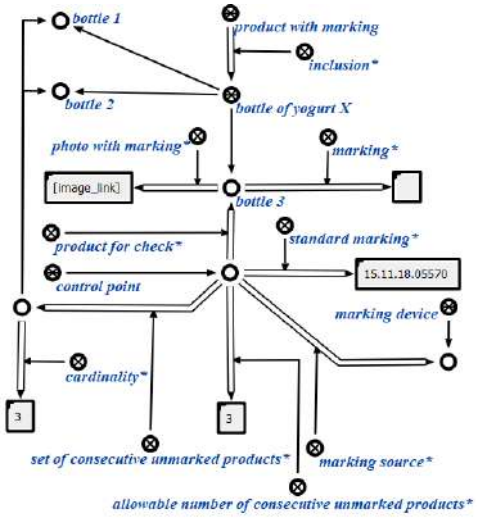


Figure 14. Fragment of the knowledge base that describes knowledge about the processed bottle after finish work of the recognition module

*then the product does not belong to a set of correctly marked products* (figure 15).

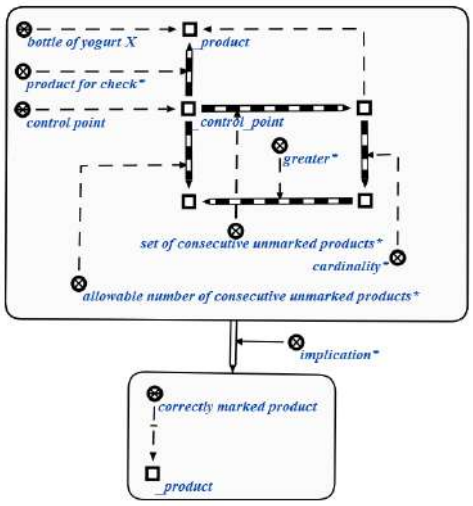


Figure 15. Logical validation rule for exceeding the allowable number of consecutive unmarked products

The solver can't use this rule because there is no information about whether a product belongs to the set of consecutive unmarked products. The solver then looks for logical rules that can provide this information and finds the following rule: *if the marking of a product is equal to the empty string, then the product belongs to the set of consecutive unmarked products*(figure 16).

This rule also cannot be applied, since the marking still needs to be compared with an empty string, for which solver uses the lexicographical string comparison program (Figure 10). Next, the solver recursively goes up the search tree and applies the logical rule from figure

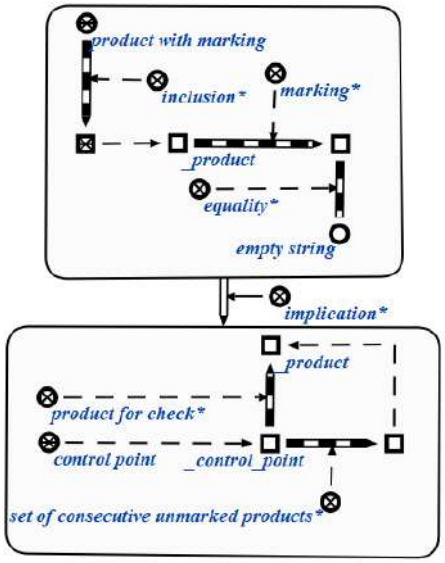


Figure 16. Logical rule of handling the situation of exceeding the allowable number of consecutive unmarked products

16 and then the logical rule from figure 15. The search for a decision is completed since the logical conclusion came to one of the two final states of the first stage.

Since the product was not added to the set of correctly marked products, the second stage begins. The problem solver starts the reverse-inference, starting with one of the final states of the second stage described in the VII section.

The first logical rule that the solver will try to apply is as follows: *if the paint level in the printer is zero, then the decision is to send the engineer reports about the lack of paint and to call the program of products rejection* (figure 17).

The solver can't apply this rule because the paint level in the printer greater than zero and it finds the following rule: *if the paint level in the printer is zero, then the decision is to send the engineer reports about the lack of paint and to call the program of the products rejection* (figure 18).

After applying this logical rule, system creates the decision node, which stores the control point, the set of message templates to be sent, and the set of programs to be called for the decision making. Next, an abstract sc-agent of using decision works with this node, which compiles and sends all messages to the engineer, as well as calls all the necessary programs. All the necessary parameters for composing messages and calling programs are taken from the control point attached to the decision node.

## CONCLUSION

The use of the considered approach to the building of decision-making systems based on the integration

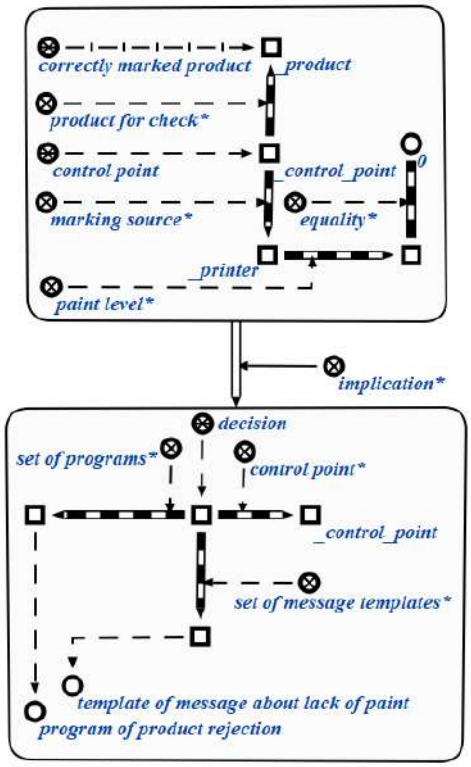


Figure 17. Logical decision rule in case of paint lack in the printer

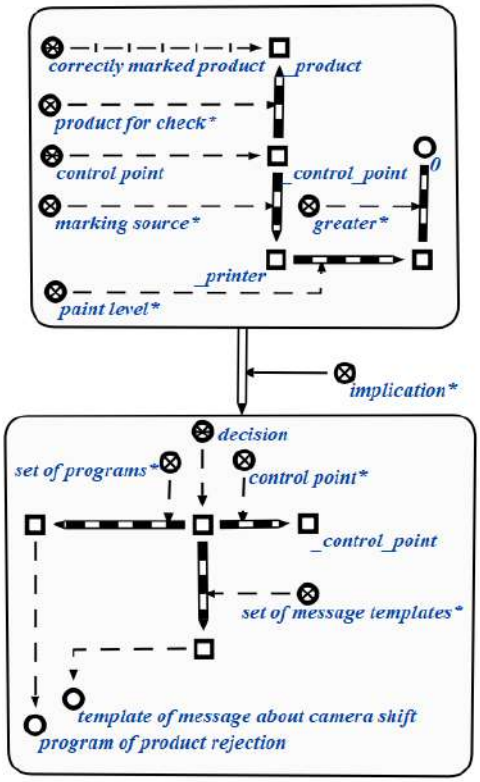


Figure 18. Logical rule of decision-making in case of camera shift

of neural network and semantic models allows us to design systems with a high level of intelligence. Such systems are able not only to make or propose decisions but also to provide its justification. However, a greater level of integration is required for a deeper retrospection of the system, in which the system will be able to analyze and justify its work not only in the search for decision but also on the recognition stage. Namely, the implementation of the neural network model in the knowledge base and its processing with the help of a problem solver.

The proposed subsystem of quality control marking for the JSC ≪Savushkin product≫ can be scaled to any product of the company with minimal changes, it will be enough only to configure the recognition module to detect marking on new products. The basic decision-making mechanism can be used in other decision-making systems, as it depends only on a set of logical rules, which is made by the engineer for each system.

## ACKNOWLEDGEMENTS

The presented results were conducted in close collaboration of research teams of the Department of Intelligent Information Technologies of Belarusian State University of Informatics and Radioelectronics, Department of Intelligent Information Technologies of Brest State Technical University and JSC ≪Savushkin product≫. Authors would like to thank Daniil Shunkevich, associate professor of the Department of Intelligent Information Technologies of Belarusian State University of Informatics and Radioelectronics, for help and valuable comments.

## REFERENCES

[1] V. Golovko, A. Kroshchanka, V. Golenkov, V. Ivashenko, M. Kovalev, V. Taberko, and D. Ivaniuk, "Integration of artificial neural networks and knowledge bases,"Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems], vol. 2, pp. 119–132, 2018.

[2] V. Golenkov and N. Gulyakina, "Proekt otkrytoi semanticheskoi tekhnologii komponentnogo proektirovaniya intellektual'nykh sistem. Chast' 2: Unifitsirovannye modeli proektirovaniya [project of open semantic technology of component design of intelligent systems. part 2: Unified design models] Ontologiya proektirovaniya [Ontology of design], no. 4, pp. 34–53, 2014, (in Russian).

[3] G. Zagorulko, Y. Zagorulko, "Podhod k organizacii kompleksnoi podderjki processa razrabotki intellektualnih SPPR v slaboformalizovannih predmetnih oblastyah Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems], vol. 2, pp. 61–64, 2016.

[4] (2018, Dec.) IMS metasystem. [Online]. Available: http://ims.ostis.net

[5] D. Pospelov, "Situacionnoe upravlenie. Teoriya i praktika Moscow: Nauka, 228p, 1986, (in Russian).

[6] V. Ivashenko, "Ontologicheskaya model' prostranstvenno vremennykh otnoshenii sobytii i yavlenii v protsessakh obrabotki znanii Vestnik BrSTU, Brest, No5(107), pp. 13—17, 2017, (in Russian).

[7] V Tarasov, "Ot mnogoagentnih sistem k intellektualnim organizaciyam filosofiya psihologiya informatika Moscow: Editorial URSS, 352 p., 2002, (in Russian).

[8] D. Shunkevich, "Agent-oriented models, method and tools of compatible problem solvers development for intelligent systems," Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems], vol. 2, pp. 119–132, 2018.

[9] E. Andreev, N. Kucevich, O. Sinenko, "SCADA-sistemy: vzglyad iznutri Moskow: RTSoft, 2004, 176 p.

[10] I. Davydenko, "Semantic models, method and tools of knowledge bases coordinated development based on reusable components" Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems], vol. 2, pp. 99–118, 2018.

[11] A. Ragovskii, "Intellektualnaya mnogoagentnaya sistema deduktivnogo vivoda na osnove setevoi organizacii Iskusstv. intellekt i prinyatie reshenii [Artificial intelligence and decision making], vol.2, pp. 73-86, 2011.

[12] V. Vagin, A. Zagoryanskaya, M. Fomina, "Dostovernii i pravdopodobnii vivod v intellektualnih sistemah Moscow: FIZMATLIT, 2008, 704 p., (in Russian).

[13] S. Yakimchik, D. Shunkevich, "Principi postroeniya reshatelei zadach v prikladnih intellektualnih sistemah Minsk, ITS 2014, pp. 160-161. (in Russian).

[14] W., Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, A. C. Berg, "SSD: Single Shot MultiBox Detector [Online]. Available: https://arxiv.org/pdf/1512.02325.pdf.

[15] A. Wong, M. J. Shafiee, F. Li, B. Chwyl, "Tiny SSD: A Tiny Single-shot Detection Deep Convolutional Neural Network for Real-time Embedded Object Detection [Online]. Available: https://arxiv.org/pdf/1802.06488.pdf.

[16] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications [Online]. Available: https://arxiv.org/pdf/1704.04861.pdf.

[17] S. Ren, K. He, R. Girshick, J. Sun "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [Online]. Available: https://arxiv.org/pdf/1506.01497.pdf.

[18] J. Hui, "Object detection: speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3) [Online]. Available: https://medium.com/@jonathan_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359.

[19] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection [Online]. Available: https://arxiv.org/pdf/1506.02640.pdf.

[20] V. Golovko, V. Krasnoproshin, "Neirosetevie tehnologii obrabotki dannih: uchebnoe posobie Minsk: BSU, 263 p., 2017, (in Russian).

[21] K. Simonyan, A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition [Online]. Available: https://arxiv.org/pdf/1409.1556.pdf.

[22] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition [Online]. Available: https://arxiv.org/pdf/1512.03385.pdf.

[23] V. Golovko, A. Kroshchanka, D. Treadwell, "The Nature of Unsupervised Learning in Deep Neural Networks: A New Understanding and Novel Approach Optical Memory And Neural Networks (Springer Link), 2016, Vol. 25, № 3, pp. 127—141.

[24] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors Computer Vision and Pattern Recognition, 2017, pp. 7310–7319.

[25] J. Hui, "What do we learn from single shot object detectors (SSD, YOLOv3), FPN & Focal loss (RetinaNet)? [Online]. Available: https://medium.com/@jonathan_hui/what-do-we-learn-from-single-shot-object-detectors-ssd-yolo-fpn-focal-loss-3888677c5f4d.

[26] K. Oksuz, B. C. Cam, E. Akbas, S. Kalkan, "Localization Recall Precision (LRP): A New Performance Metric for Object Detection [Online]. Available: https://arxiv.org/pdf/1807.01696.pdf.

## ПРИНЦИПЫ ПОСТРОЕНИЯ СИСТЕМ ПРИНЯТИЯ РЕШЕНИЙ НА ОСНОВЕ ИНТЕГРАЦИИ НЕЙРОСЕТЕВЫХ И СЕМАНТИЧЕСКИХ МОДЕЛЕЙ

Головко В. А., Крощенко А. А., Таберко В. В., Иванюк Д. С., Ивашенко В. П., Ковалев М. В.

В работе рассматриваются преимущества интеграции нейросетевых и семантических моделей для построения систем принятия решений. Предложен подход к интеграции искусственных нейронных сетей с базами знаний по входам и выходам и спецификация этих сетей в базе знаний с использованием онтологий соответствующих предметных областей. Предложенный подход рассматривается на реальных производственных задачах ОАО «Савушкин продукт» для контроля качества маркировки продукции.

Received 29.12.18