

Algorithm for constructing logical operations to identify patterns in data.

Lyutikova L.A. and Shmatova E.V. (lylarisa@yandex.ru).

Institute of Applied Mathematics and Automation KBSC RAS (IAMA KBSC RAS), Nalchik, Russia.

ABSTRACT

The paper proposes an algorithm for constructing logical operations, in terms of multi-valued logic, to identify hidden patterns in poorly formalized areas of knowledge. As the basic elements are considered many functions of the multi-valued logic of generalized addition and multiplication. The combination of these functions makes it possible to detect relationships in the data under study, as well as the ability to correct the results of neural networks. The proposed approach was considered for classification problems, in the case of multidimensional discrete features, where each feature can take k -different values and is equivalent in importance to class identification.

Formulation of the problem

Consider a multi-valued logical system $L = \langle L, \bar{x}, \&, \vee, \bar{\cdot}, \rightarrow \rangle$, $L = \{0, 1, \dots, k-1\}$.

Definition. Set of functions $\sigma(x, y)$: such that $\sigma(x, 0) = \sigma(0, x) = x$, we will call functions of generalized addition.

To construct the decisive function $Y = f(X)$, it is required to find a set of functions Σ that satisfy the following conditions: $\Sigma(a_1, \dots, a_{n-1}, a_n) = y$.

The algorithm is built in the form of a tree.

We assume that $\Sigma(a_1, \dots, a_{n-1}, a_n) = \Sigma(a_1, \dots, a_{n-2}, a_n) + a_n = \sigma(\Sigma(a_1, \dots, a_{n-2}, a_n), a_n)$. Great $\Sigma(a_1, \dots, a_{n-1})$ suspense, but it must take on one of the meanings $L = \{0, 1, \dots, k-1\}$. This implies the need to fulfill one of k conditions:

$$\Sigma(a_1, \dots, a_{n-1}) = 0, \Sigma(a_1, \dots, a_{n-1}) = 1, \dots, \Sigma(a_1, \dots, a_{n-1}) = k-1 \quad (1)$$

Means

$$\Sigma(0, a_n) = y, \Sigma(1, a_n) = y, \dots, \Sigma(k-1, a_n) = y, \quad (2)$$

those. get to the branching tree.

In the next step, we consider the following relation $\Sigma(a_1, \dots, a_{n-1}) = \Sigma(a_1, \dots, a_{n-2}, a_{n-1}) + a_{n-1} = \sigma(\Sigma(a_1, \dots, a_{n-2}, a_{n-1}), a_{n-1})$, taking into account the previously formulated assumptions regarding the values of $\Sigma(a_1, \dots, a_{n-1})$, i.e. each of the branches (2) will in turn split into another k .

Continuing to carry out the steps described above, we construct a tree of admissible values of the truth tables Σ .

If at some step the assumption contradicts the assumption made earlier regarding the given variant of the function Σ , then such a branch is a dead end and it is discarded.

If at some node all branches are dead ends, then this node itself is deleted from the decision tree.

The last step in these actions is to consider the expression $\Sigma(a_1, a_2)$, after which the process of finding a solution ends.

The set of feasible solutions of the function Σ is collected from the leaves of the tree to the root. As a result, we obtain truth tables of the function Σ that satisfy the given conditions.

We illustrate the operation of the algorithm using the example of three-valued logic.

An algorithm for constructing a decisive function

The object will be represented by n -dimensional vector, n - the number of characteristic features of the object in question, the j -th coordinate of this vector is equal to the value of the j -th characteristic, $j = 1, \dots, n$. Information about any characteristic of the object may be missing. The dimensionality of the considered property of the $k_j \in [2, \dots, N]$, N - object depends on the encoding method of the i -th characteristic [4].

Let $X = \{x_1, x_2, \dots, x_n\}$, $x_i \in \{0, 1, \dots, k_i - 1\}$, where $k_i \in [2, \dots, N]$, is a set of properties that characterizes a given object. $Y = \{y_1, y_2, \dots, y_m\}$ - many considered objects. For each object y_i there is a corresponding set of features

$$x_i(y_1), \dots, x_n(y_i): y_i = f(x_1(y_1), \dots, x_n(y_i)).$$

Or $X = \{x_1, x_2, \dots, x_n\}$, where $x_i \in \{0, 1, \dots, k_i - 1\}$, $k_i \in [2, \dots, N]$, $N \in \mathbb{Z}$ - input, $X_i = \{x_1(y_i), x_2(y_i), \dots, x_n(y_i)\}$, $i = 1, \dots, n$, $y_i \in Y$, $Y = \{y_1, y_2, \dots, y_m\}$ - output:

$$\begin{pmatrix} x_1(y_1) & x_2(y_1) & \dots & x_n(y_1) \\ x_1(y_2) & x_2(y_2) & \dots & x_n(y_2) \\ \dots & \dots & \dots & \dots \\ x_1(y_m) & x_2(y_m) & \dots & x_n(y_m) \end{pmatrix} \rightarrow \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{pmatrix}$$

It is necessary to construct a function such that $Y = f(X)$.

A function $Y = f(X)$ is called a decisive function.

The dependence under consideration can be approximated using a neural network built on the basis of elements that implement external summation and a continuous scalar function.

$$sp(x_1, \dots, x_n) = \sum w_i \prod x_i$$

where $\{w_1, w_2, \dots, w_k\}$ is the set of weights of a given $\Sigma\Pi$ neuron that recognizes k elements of a given subject area $Y = \{y_1, y_2, \dots, y_k\}$ formed by a corresponding set of features $\{X_1, \dots, X_k\}$.

Of great interest are direct sums that allow you to simultaneously form the architecture of a computer network and configure its parameters, without resorting to solving complex optimization problems to achieve the correctness of its functioning [5].

Let three-valued logic be given $L = \langle L, \bar{x}, \&, \vee, \bar{\cdot}, \rightarrow \rangle$, $L = \{0, \frac{1}{2}, 1\}$. And let it be given: $\Sigma(1, 0, \frac{1}{2}, 1) = \frac{1}{2}$. We construct a decision tree for this example.

Consider 1 branch: $\Sigma(1, 0, \frac{1}{2}) = 0 \Rightarrow$ it is necessary that $\Sigma(0, 1) = \frac{1}{2}$, but this contradicts the condition $\sigma(x, 0) = \sigma(0, x) = x$. So this branch can be discarded.

Consider a 2 branch $\Sigma(1, 0, \frac{1}{2}) = \frac{1}{2} \Rightarrow \Sigma(\frac{1}{2}, 1) = \frac{1}{2}$. This is possible, therefore we build branches further:

1. $\Sigma(1, 0) = 0$ - impossible, therefore, the branch can be dropped;

2. $\Sigma(1, 0) = \frac{1}{2}$ - impossible, therefore, the branch can be dropped;

3. $\Sigma(1, 0) = 1$ - possible.

Restoring the chain of actions, we obtain: $\Sigma(1, 0) = 1$, $\Sigma(\frac{1}{2}, 1) = \frac{1}{2}$, $\Sigma(\frac{1}{2}, 1) = \frac{1}{2}$.

From this we can conclude: that the obtained set of solutions (we will call the class of solutions) has the commutativity property (Table 1).

Consider the 3-rd branch: $\Sigma(1, 0, \frac{1}{2}) = 1 \Rightarrow \Sigma(1, 1) = \frac{1}{2}$. This is perhaps why we build the branches further:

1. $\Sigma(1, 0) = 0$ - impossible, therefore, the branch can be dropped;

2. $\Sigma(1, 0) = \frac{1}{2}$ - impossible, therefore, the branch can be dropped;

3. $\Sigma(1, 0) = 1$ - possible. Restore the chain of actions, we get:

$\Sigma(1, 0) = 1$, $\Sigma(\frac{1}{2}, 1) = 1$, $\Sigma(1, 1) = \frac{1}{2}$. Those. got the truth table

without taking into account the commutativity property (tab. 2). If commutativity is taken into account, then we get Table 3.

Table 1.

	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$		$\frac{1}{2}$
1	1	$\frac{1}{2}$	

Table 2.

	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$		1
1	1		$\frac{1}{2}$

Table 3.

	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$		1
1	1	1	$\frac{1}{2}$

Algorithm for constructing a decision function based on the generalized multiplication operation

When solving problems of constructive learning, there is a need to find functions that most effectively implement the specified training samples.

Let the input of the system be fed a vector of values $x = \{x_1, x_2, \dots, x_n\}$ and each input has the weight w_j , $j = 1, \dots, m$, the output of the system has the resulting offset y .

You need to build (set in a table) a set of functions that satisfy the condition: $f(x_i, w_i) = y$, where $f(x_i, w_i)$ is a function that can be represented through the operations of generalized addition and generalized multiplication.

Let three-valued logic be given $L = \langle L, \bar{x}, \&, \vee, \bar{\cdot}, \rightarrow \rangle$, $L = \{0, 1, 2\}$.

Definition 2: Many functions $\pi(x, y)$: $\pi(x, 0) = \pi(0, x) = 0$ we call the implementation of generalized multiplication.

For given input values x and w and output y , we have a partially defined three-digit function, which is defined on the set (x, w) by the value y .

For this, we first construct the admissible operations of generalized multiplication for x_i, w_i , $i = 1, 2, \dots, n$ in the form of a tree.

Consider x_i, w_i . This quantity is unknown to us, but must take one of the values $L = \{0, 1, 2\}$ (due to the closedness of the operation of generalized multiplication).

This implies the need to fulfill one of three conditions:

$$x_i w_i = 0, x_i w_i = 1, x_i w_i = 2.$$

In this case, we obtain three possibilities for implementing the function $\Pi \in \pi$ (i.e., three tree branches).

At the next step, we consider the following relation $x_2 w_2$ and, taking into account the previously formulated assumptions, each of the branches (2) will in turn split into three branches.

Continuing to carry out the steps described above, we construct a tree of admissible values of the truth tables of the operation of generalized multiplication.

If at some step the assumption contradicts the earlier assumption regarding this variant of the function Π , then such a branch is dead-end and it is discarded.

The last step is to consider $x_n w_n$, after which the process of finding generalized multiplication ends.

The set of feasible solutions to Π is collected by lifting from the top leaves of the tree to the root. As a result, we obtain a set of truth tables of the function of generalized multiplication.

If the collection of the set π is empty, then for the functions $= \{x_1, x_2, \dots, x_n\}$, $w = \{w_1, w_2, \dots, w_n\}$ one cannot specify the operations of generalized addition and generalized multiplication so that relation (1) holds.

Let $a_i = x_i w_i$, $i = 1, \dots, n$, then each found operation $\Pi \in \pi$ can be associated with a vector $a = \{a_1, a_2, \dots, a_n\}$. Those. (1) rewritten: $\Sigma(a_1, a_2, \dots, a_n) = y$.

And then, to determine the operation of generalized multiplication, we use the algorithm of the representable function through the operations of generalized addition, proposed in the previous section.

CONCLUSIONS

It can be argued that the logical functions obtained as a result of the proposed algorithms are adequate solvers of the task and make it possible to identify hidden patterns in the data.

ACKNOWLEDGMENTS & REFERENCES

The reported study was funded by RFBR according to the research project №18-01-00050-a