Artificial Neural Networks that Change their Configuration

Tatiana Kosovskaya
Sankt-Petersburg State University
Sankt-Petersburg, Russia
e-mail: kosovtm@gmail.com

https://doi.org/10.62343/csit.2024.8

***Abstract* — An analogue of a neural network, the number of layers and the number of cells in which may be changed during its retraining is suggested in the paper. The main instrument for constructing such a network is extraction of maximal common properties of pairs of objects in the training set and of that ones used for retraining. The degree of coincidence of a recognized object with the one presented in the training set may be calculated using their maximal common properties. Computational complexities of such a network construction, recognition process and the network retraining are proved. A brief description of a similar network proposed by the author earlier for complex structured objects described using predicate calculus is presented. The analysis of comparison of computational complexity of a complex structured object recognition with various methods of their description is given.**

***Keywords* — neural network; maximal common property of objects; degree of coincidence; computational complexity.**

I. INTRODUCTION

The origins of the appearance of artificial neural networks lie in the works of F. Rosenblatt, dedicated to the creation of a perceptron, a machine for solving pattern recognition problems [1]. However, the significant disadvantages of the perceptron led to the creation of other models for solving such problems. One of the most common models currently is a neural network.

An artificial neuron is only a model of a neuron in a living organism. Therefore, it will be called a cell below. At the same time, the contents of such a cell may vary depending on the way the problem is formalized.

When creating a modern artificial neural network, the researcher sets its configuration in advance: the number of network layers and the number of cells in each layer. This does not correspond to how a neural network is built in the brain of a living organism, in which new neurons are added during the learning process, some connections are broken and new ones are formed. Besides, classical artificial neural networks have the disadvantage that unpredictable "outliers" sometimes occur while recognition new objects. Results that are poorly explained from a theoretical point of view.

In this regard, various modifications of artificial neural networks began to be developed.

Overfitting is a serious problem in such networks. For example, a dropout network was suggested in [2] to overcome such a problem. To detect regularities in datasets, polynomial neural networks were developed [3], [4]. Due to the huge volumes of processed data, fuzzy neural networks have found their application.

In this paper, it is proposed to construct an analogue of a neural network, the number of layers and the number of cells in which may be changed during its retraining.

The basis for the construction of the proposed networks is the concept of the maximum common property (MCP) of objects. This concept was introduced earlier by the author for complex structured objects described using predicate formulas [5]. The problem of finding MCP of such objects has expo-nential complexity [6].

For objects described in the terms of binary or finite-valued features, the problem of finding MCP is a polynomial one. Such a difference in computational complexity depending on the chosen description language is discussed in the Discussion section of the paper.

II. PRE-TRAINING THE NETWORK FOR OBJECTS DESCRIBED BY FINITE VALUED FEATURES

A. Setting of Problem

Let Ω be a set (potentially infinite) of objects. A set of features $p_1, \ldots, p_n$ that define the description of an object ω from Ω in the form of a string of values of these features $\alpha = (\alpha_1, \ldots, \alpha_n)$ is defined for objects under study. A training set (TS) $\{\omega^1, \ldots, \omega^K\}$ of objects from Ω with descriptions $\alpha^k = (\alpha_1^k, \ldots, \alpha_n^k)$, $k = 1, \ldots, K$ is given.

It is required to construct a network that gives the answer "YES" for objects from TS. If an object is absent in TS we can not guarantee its correct recognition. Fuzzy recognition of such an object consists in calculation of maximal degree of coincidence with one of the objects from TS.

B. Important Definitions

Definition 1. *The maximal common property (MCP) of objects with descriptions* $\alpha^k = (\alpha_1^k, \ldots, \alpha_n^k)$ *and* $\alpha^m = (\alpha_1^m, \ldots, \alpha_n^m)$ *is a string of the form* $\alpha^{km} = (\alpha_1^{km}, \ldots, \alpha_n^{km})$, *where*

if $\alpha_i^k \neq \alpha_i^m$ *then* $\alpha_i^{km} = *$,
if $\alpha_i^k = \alpha_i^m$ *then* $\alpha_i^{km} = \alpha_i^k$.

C. Initial Network Training

Network cells with zero out-degree contain descriptions of objects from TS.

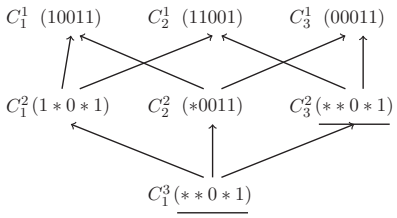

Fig. 1. The result of pairwise extractions of MCPs.

For each pair of objects from the TS $\alpha^k = (\alpha_1^k, \ldots, \alpha_n^k)$ and $\alpha^m = (\alpha_1^m, \ldots, \alpha_n^m)$ we find their MCP α^{km}. Cells with α^{km} and α^k and α^m are connected by oriented edges. If some pairs of objects have the same MCPs, then their corresponding cells are identified.

Repeat the process with the already extracted MCPs. At the same time, if on the lth repetition we received the same MCP that was received earlier, then the corresponding cells of the network are identified.

The process will stop as the MCP lengths decrease.

D. Computational complexity of initial network training

Let K be the number of objects in the TS.

When the MCPs for pairs of descriptions are found for the first time, $K_1 = \frac{K(K-1)}{2}$ extractions of MCP are produced.

When the MCP for pairs of descriptions obtained on the $l-1$-th layer is found for the l-th time, no more than $\frac{K_l(K_l-1)}{2}$ MCP extractions, each of which requires linear of less (at least by l) than n steps. That is, the $l-1$-th extraction requires $O(K^{2^l}n)$ steps.

At the same time, l does not exceed n. Thus, the total number of steps of initial network training will be $O(\sum_{l=1}^{n-1} K^{2^l}n) = O(K^{2^n})$.

The resulting estimate is very large and achievable.

E. Example of initial network construction

Let the objects be described by five binary features and a TS of three elements with descriptions (10011), (11001) and (00011) is given.

According to the described above algorithm the obtained network has the form presented in Figure 1.

The cells C_3^2 and C_1^3 must be glued because they contain the same MCP. The result of gluing the cells C_3^2 and C_1^3 is presented in Figure 2.

It must be noted that now we can not say what is the number of level to which the cell C_1^3 belongs.

III. RECOGNITION PROCESS

Obviously, such a network can precisely recognize only such an object whose description coincides with the descrip-tion of some object from the TS. The process of such a recognition for an object with the description (11011) is presented in Figure 3.

Let an object (which was previously absent in the TS) with the description $\alpha = (\alpha_1, \ldots, \alpha_n)$ be presented for recognition.

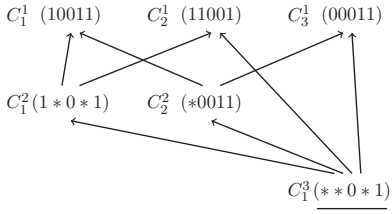

Fig. 2. The result of gluing the cells C_3^2 and C_1^3.

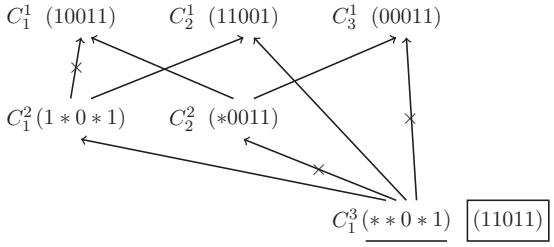

Fig. 3. Recognition of an object with description presented in TS.

At each check of whether the cell contents match its description, in case of a negative result, the degree of their coincidence deg_C is calculated in the cell C. The degree of an object description coincidence with the contents of a cell is the ratio of the number of matched values to the length of the contents. Note that the length and match of the contents is calculated without taking into account the number of occurrences of the $*$ symbol. For example, if the description of a recognized object is (10001) and the contents of a cell C is $(*0011)$ then $deg_C = \frac{3}{4}$. For a cell C' with the contents $(1*0*1)$ $deg_{C'} = 1$.

Why do we calculate the degree of coincidence in the intermediate cells? It is not necessary. But it can allow us to make a traversal of the network beginning with the cells with the largest degree of coincidence.

Let an object with the description (10001) be presented for recognition. This object was absent in the TS.

The process of recognizing an object with a description (10001) that was absent in TS is shown in Figure 4.

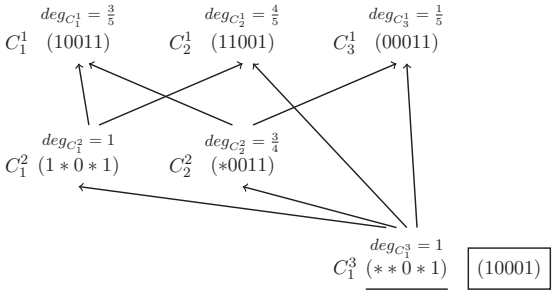

Fig. 4. Recognition of an object with description absent in TS.

A. Computational complexity of recognition process

Since the process of recognition of an object is reduced to the graph traversal from the root cells to the leafs comparing at each cells of strings with the length no more than the number of features, the estimate (very rough) for the number of steps of the recognition process will be

$$O((N+M)n),$$

where N is the number of cells in the network, M is the number of edges in the network ($M < N^2$), n is the number of features, in terms of which the description of the object is obtained.

But if the recognized object was presented in TS then its recognition will be done in $O(h \cdot n)$ number of steps, where h is the "height" of the network. Since $h < n$, this bound is $O(n^2)$.

IV. NETWORK RETRAINING

An already constructed network can be retrained on a new object with the description $\alpha = (\alpha_1, \ldots, \alpha_n)$, about which it is known that the network should give an answer "YES". To do this, it is enough, firstly, to find the MCP of the presented object with the contents of the leafs. Secondly, to find the MCP of the contents of newly received cells with the contents of the old ones. And finally, to glue cells with the same contents.

As a result of this process, new cells may appear in the network, and both the number of layers and the number of cells in the network may be changed.

A. Computational complexity of network retraining

Let N be the number of cells in the network before retraining, N_0 be the number of leafs, N_1 be the number of cells having oriented edge ending in leafs, ..., N_h be the number of roots. Note, that $h < n$ and $N_0 + N_1 + \cdots + N_h = N$.

The first MCP extractions from the description of a new object and the contents of leafs requires $N_0 n$ steps. These extractions can give no more than N_0 new cells.

Next extractions can require $N_1 \cdot N_0 \cdot n$, $N_2 \cdot N_1 \cdot N_0 \cdot n$, ..., $N_h \ldots N_2 \cdot N_1 \cdot N_0 \cdot n$ steps. The total number of steps is $O(N_h \ldots N_2 \cdot N_1 \cdot N_0 \cdot n)$.

So, we have an exponential upper bound for network retraining number of steps. For example, if $N_1 = \cdots = N_h = \frac{N}{n}$ then computational complexity is $O((\frac{N}{n})^n)$.

V. NETWORK FOR COMPLEX STRUCTURED OBJECTS

The idea of fuzzy neural network that changes its configuration came to the author while investigating complex structured object recognition. It is convenient to describe such objects in the terms of predicate formulas [7], [8]. In such a case, recognition problems turn out to be NP-complete or NP-hard [6]. To decrease computational complexity of algorithms solving such problems, the author earlier has introduced notions of level description of objects, networks changing their configuration [9] and fuzzy logic-predicate networks [5].

This section will give a brief description of this approach. Then the difference in computational complexity of the two approaches will be discussed in the Discussion section.

A. Problem setting

Let an investigated object ω be a complex structured object, which is presented as a set of its elements $\omega = \{\omega_1, \ldots, \omega_t\}$ and is characterized by predicates $p_1, \ldots, p_n$. These predicates define some properties of its elements or relations between them. The description $S(\omega_1, \ldots, \omega_t)$ of the object ω is a set of all constant literals (atomic formulas or their negations) with predicates $p_1, \ldots, p_n$ which are true for ω.

The set of all objects Ω is divided into K classes $\Omega = \Omega_1 \cup \cdots \cup \Omega_K$. Formula $A_k(\overline{x}_k)$ have the form of disjunction of elementary conjunctions and is true if $\omega \in \Omega_k$.

The recognition of ω consists in checking

$$S(\omega) \Rightarrow \exists \overline{x}_{k \neq} A_k(\overline{x}_k). \quad (1)$$

To denote that there exist distinct values for variables from the list of variables $\overline{x}$ the notation $\exists \overline{x}_{\neq} A_k(\overline{x})$ is used.

This is an NP-complete problem. Depending on an algorithm searching for the values of $\overline{x}_k$ in (1), computational complexity is exponential on the number of literals in $A_k(\overline{x}_k)$ or on the number of arguments in it [6].

If we need not only to check the formula (1), but to find values for $\overline{x}_k$ then it becomes NP-hard. Nevertheless, both an exhaustive search algorithm and an algorithm based on derivation in predicate calculus not only answer the question "if their exists?" but find values for $\overline{x}_k$.

B. Level description of classes

Definition. *Elementary conjunction $R(\overline{z})$ is called an MCP of two elementary conjunctions $A(\overline{x})$ and $B(\overline{y})$ if it is their maximal common up to the names of arguments sub-formula.*

Instead of the term "common up to the names of arguments sub-formula" the term "isomorphic" is used in [5].

While MCP extraction for objects described by a binary (or multi-valued) string is a polynomial-in-time procedure, MCP extraction for complex structured objects represented as a set of its elements and described by a predicate formula is NP-hard problem [6].

To decrease computational complexity of checking (1), it was suggested in [5] to extract pairwise MCPs $P_1^L(\overline{y}_1^L), \ \ldots, \ P_{n_L}^L(\overline{y}_{n_L}^L)$ for disjunctive terms of $A_k(\overline{x}_k) = A_{k,1}(\overline{x}_{k,1}) \vee \cdots \vee A_{k,m_k}(\overline{x}_{k,m_k})$.

Simultaneously new unary predicates $p_i^L(y_i^L)$ and new variables y_i^L for lists of initial variables $\overline{y}_i^L$ defined by equalities $p_i^L(y_i^L) \Leftrightarrow P_i^L(\overline{y}_i^L)$ are introduced.

Repeat this procedure with formulas $P_1^l(\overline{y}_1^l), \ \ldots, \ P_{n_l}^l(\overline{y}_{n_l}^l)$ for $l = L, \ldots, 2$ and receive $P_1^{l-1}(\overline{y}_1^{l-1}), \ \ldots, \ P_{n_{l-1}}^{l-1}(\overline{y}_{n_l}^{l-1})$.

Every occurrence of $P_i^l(\overline{y}_i^l)$ in $P_{j'}^{l'}(\overline{y}_{j'}^{l'})$ ($l' > l$) and in $A_{k,j}(\overline{x}_{k,j})$ replace by $p_i^l(y_i^l)$.

Let $A_{k,j}^L(\overline{x}_{k,j}^L)$ be the results of substitutions of $p_i^l(y_i^l)$ instead of $P_i^l(\overline{y}_i^l)$ into $A_{k,j}(\overline{x}_{k,j})$. Level description of formulas $A_{k,1}(\overline{x}_{k,1}), \ldots, \ A_{k,m_k}(\overline{x}_{k,m_k})$ has the form (2) [5].

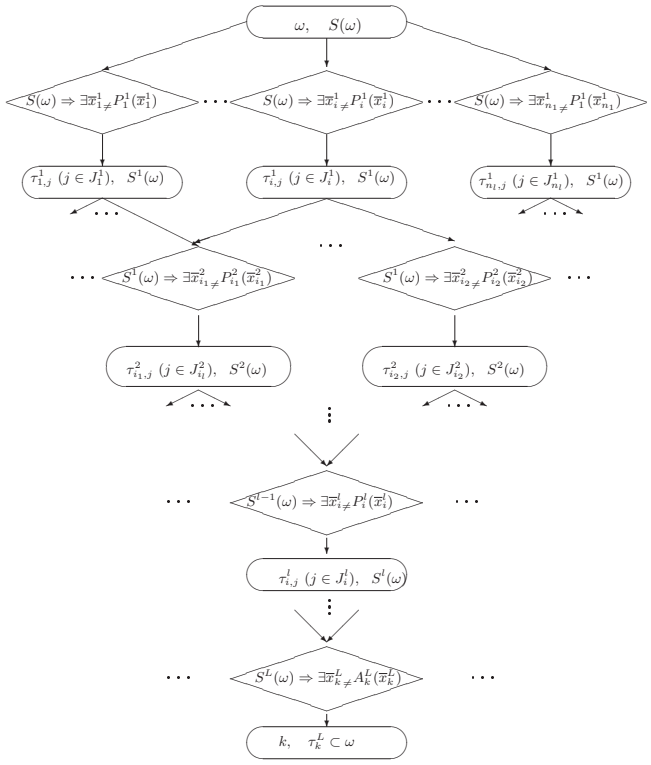

Fig. 5. Process of level recognition.

$$\begin{cases} & A^L_{k,j}(\overline{x}^L_{k,j}) & (j=1,\ldots,m_k) \\ p^1_1(y^1_1) & \Leftrightarrow & P^1_1(\overline{y}^1_1) \\ & \vdots & \\ p^1_{n_1}(y^1_{n_1}) & \Leftrightarrow & P^1_{n_1}(\overline{y}^1_{n_1}) \\ & \vdots & \\ p^l_i(y^l_i) & \Leftrightarrow & P^l_i(\overline{y}^l_i) \\ & \vdots & \\ p^L_{n_L}(y^L_{n_L}) & \Leftrightarrow & P^L_{n_L}(\overline{y}^L_{n_L}) \end{cases} . \qquad (2)$$

C. Level recognition

The process of level recognition is presented in Figure 5. (This figure was published in the author's paper [5].)

Here we can see why the author uses the term "cell" instead of the term "neuron". The contents of cells in Figure 5 are elementary conjunctions of predicate formulas. Every condition which is checked in a rhomb of this scheme has the same form as the formula (1). But its right-hand part is essentially shorter than that in (1). That is why computational complexity essentially decreases.

But, nevertheless, the problem remains NP-complete (NP-hard if we need to find values of arguments in (1)).

VI. DISCUSSION

It is well known that every information may be represented in the form of a binary string.

The main part of this paper is devoted to the recognition problem of objects described by binary strings. It was shown that computational complexity of recognition of an object with description presented in TS is $O(n^2)$, where n is the number of features (i.e., the length of description).

In section 6, it was mentioned (with reference to [5]) that computational complexity of recognizing a complex structured object is an NP-complete problem.

Is it a fraud? Or it was proved that **P=NP**?

Computational complexity is a function of the input data length.

Let $\omega = \{\omega_1, \ldots, \omega_t\}$ and be characterized by predicates $p_1, \ldots, p_n$. The length of a binary string which simulates an elementary conjunction of predicate formulas with t variables and containing an m-ary predicate is $O(t^m$

It corresponds to the the estimate $O(t^m \cdot |A| \cdot |S|)$ given in [7] for the number of propositional variables modelling predicate calculus formulas in a finite domain.

As polynomial of the exponent gives the exponent, we have not prove that **P=NP**. And it was not a fraud that the same problem with different representations of input data belongs to essentially different complexity classes.

ACKNOWLEDGMENT

The author acknowledges Saint-Petersburg State University for a research project 95438429

REFERENCES

1. F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain." Psychological Review, vol. 65, no. 6, pp. 386–408, 1958.
2. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." J. of Machine Learning Research, no. 15, pp. 1929 1958, 2014.
3. G.G. Chrysos, S. Moschoglou, G. Bouritsas, Y. Panagakis, J. Deng and S. Zafeiriou, "P-nets: Deep Polynomial Neural Networks." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), China, pp. 7325–7335, 14 06 2020.
4. A. Zou, R. Deng, Q. Mei, et al, "Fault diagnosis of a transformer based on polynomial neural networks." Cluster Comput, vol. 22, pp. 9941–9949, 2019.
5. T. Kosovskaya, "Fuzzy logic-predicate network." In Proceedings of the 11th Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (EUSFLAT 2019), Prague, Czech, 9-13 09 2019, pp. 9 – 13.
6. T. Kosovskaya, Predicate Calculus as a Tool for AI Problems Solution: Algorithms and Their Complexity. Chapter 3 in Intelligent System. Open access peer-reviewed Edited volume. Edited by Chatchawal Wongchoosuk Kasetsart University, pp. 1 – 20, 2018. [Online]. Available: https://www.intechopen.com/chapters/58698
7. S. Russell and P. Norvig, Artificial Intelligence A Modern Approach, 4th ed.; Pearson Series in Artificial Intelligence, 2021
8. N.J. Nilsson, Problem-Solving Methods in Artificial Intelligence; McGrow-Hill Book Company, New York, 1971.
9. 9 T. Kosovskaya, "Self-modificated predicate networks." International Journal on Information Theory and Applications, vol. 22, no 3, pp. 245–257, 2015