ATLANTIS PRESS

Atlantis Studies in Uncertainty Modelling, volume 1

11th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT 2019)

Fuzzy logic-predicate network

Tatiana Kosovskaya
Prof. of St. Petersburg State University,
University emb. 7/9, St. Petersburg, 199034, Russia,
kosovtm@gmail.com

Abstract

In many Artrificial Intelligence problems an investigated object is considered as a set of its elements $\{\omega_1, \ldots, \omega_t\}$ and is characterized by properties of these elements and relations between them. These properties and relations may be set by predicates $p_1, \ldots, p_n$. The problems appeared with such an approach become NP-complete or NP-hard ones. To decrease the computational complexity of these problems a hierarchical multi-level description of classes was suggested. A logic-predicate recognition network may be constructed according to such a multi-level description. Such a network recognizes only objects which have been presented in the training set, but it may be easily retrained by a new object. After retraining it may change its configuration i.e., the number of levels and the number of nodes in every level. A modification of such a network is offered in this paper. This modification allows to do a fuzzy recognition of a new object and to calculate the degree of certainty that this object or its part belongs to some class of objects.

Keywords: Hierarchical description, logic-predicate recognition network, fuzzy recognition.

1 Introduction

The use of various networks for solving AI problems is widespread. Such networks include, for example, neural networks [1], Bayesian networks [10, 12]. All these networks allow to solve problems with a sufficient high degree of confidence.

Descriptions of objects recognized by such networks, as a rule, are a given-length strings of constants. In many cases, these descriptions can be simulated by binary strings. Processing of binary strings, as a rule, requires a linear (or at least polynomial) under the length of input data time.

This seems to be very effective in terms of the time spent on solving the problem. However, in many cases the length of such a binary string depends exponentially on the length of the problem's input data [11]. In addition, the original structure of the object under investigation is lost.

Many Artificial Intelligence problems permit their formalization by means of predicate calculus language. In such a case, an investigated object is represented as a set of its elements, and its description is a set of literals (atomic formulas or their negations) with predicates defining properties of these elements or relations between them [9]. Thus formulated problems become NP-complete or NP-hard ones [5]. The computational complexity of such problems, when being solved by an exhaustive search algorithm, coincides with the length of their encoding using a binary string [11].

To decrease the computational complexity of these problems a hierarchical level description of classes was suggested in [3]. To construct such a level description, "frequently appeared" sub-formulas of "small complexity" are extracted from class descriptions. This allows to decompose the main problem into a series of similar problems with input data with the smaller length.

Construction of a level logic-predicate network with the use of level description of classes was offered in [7].

The construction of such a network is based on the extraction from the descriptions of classes of objects of such fragments, which are inherent in many objects of classes. Recognition itself is reduced to the sequential solution of problems of the same type with a small length of the input data.

A modification of the logic-predicate network is offered in the paper. The degree of coincidence for descrip-

tion of an object part and the formula, satisfiability of which is checking in the cell, is calculated.

2 Logic-predicate approach to AI problems

A logic-predicate approach to AI problems and algorithms of their solutions in the frameworks of this approach are in [9]. Here only a problem setting and main methods of its solution are described.

Let an investigated object be represented as a set of its elements $\omega = \{\omega_1, \ldots, \omega_t\}$. A collection of predicates $p_1, \ldots, p_n$ characterizing properties of elements from ω and relations between them is given. Logical description $S(\omega)$ of the object ω is the set of all satisfiable on ω literals.

The set of all objects is partitioned on the classes $\Omega = \cup_{k=1}^{K} \Omega_k$. Logical description of the class Ω_k is a formula $A_k(\overline{x_k})$ in the form of disjunction of elementary conjunctions such that if $A_k(\overline{\omega})$ then $\omega \in \Omega_k$.[1] Note that $S(\omega)$ describes the **set** ω which is not ordered in contradistinction to the arguments of the formula $A_k(\overline{x}_k)$ or $A_k(\overline{\omega})$.

With the use of the described notations some problems may be formalized in the following way [5, 9].

Identification problem. To find all parts of the object ω which belong to the class Ω_k.[2]

$$S(\omega) \Rightarrow \exists \overline{x_k}_{\neq}\ A_k(\overline{x_k}) \tag{1}$$

Classification problem. To find all such class numbers k that $\omega \in \Omega_k$.

$$S(\omega) \Rightarrow \bigvee_{k=1}^{K} A_k(\overline{\omega}) \tag{2}$$

Formula (2) may be rewritten as

$$S(\omega) \Rightarrow \bigvee_{k=1}^{K} \exists \Pi(\omega) A_k(\Pi(\omega)), \tag{2'}$$

$\Pi(\omega)$ be a permutation of the set ω elements.

Analysis problem. To find and classify all parts τ of the object ω.

$$S(\omega) \Rightarrow \bigvee_{k=1}^{K} \exists \overline{x_k}_{\neq}\ A_k(\overline{x_k}) \tag{3}$$

[1] Here and below $\overline{x}$ is a denotation of an ordered list of a finite set x elements. If elements of the list $\overline{x}$ belong to the set y then we will write $x \subseteq y$.

[2] To mark that all values of variables from the list $\overline{x}$ satisfying the formula $A(\overline{x})$ are distinct, instead of the formula $\exists x_1 ... \exists x_m (\&_{i=1}^{m} \&_{j=i+1}^{m} (x_i \neq x_j)\ \&\ A(x_1, ..., x_m))$ the formula $\exists \overline{x}_{\neq} A(\overline{x})$ will be used.

Note that the computational complexity of the problems (1) and (3) coincides up to a multiplicative constant with the computational complexity of the following problem

$$S(\omega) \Rightarrow \exists \overline{x}_{\neq}\ A(\overline{x}), \tag{4}$$

where $A(\overline{x})$ is an elementary conjunction of atomic formulas.

The problems (1) and (3) are NP-complete [5, 9]. And the problem (2) is polynomially equivalent to the problem of graph isomorphism, which is a so called "open" problem [2], for which it is not known either it is in the class **P** either it is NP-complete.

Let's give a definition which is very important for the following.

Definition 1. Two elementary conjunctions of atomic formulas P and Q are isomorphic *if there exists such an elementary conjunction R and substitutions of arguments $a_1, \ldots, a_m$ and $b_1, \ldots, b_m$ of the formulas P and Q, respectively, instead of all occurrences of variables $x_1, \ldots, x_m$ of the formula R that the results of these substitutions into R coincide with the formulas P and Q, respectively, up to the order of literals.*

The substitutions $(a_1 \to x_{i_1}, , a_m \to x_{i_m})$ and $(b_1 \to x_{j_1}, , b_m \to x_{j_m})$ are called unifiers of formulas P and Q with the formula R respectively.

In fact, the notion of elementary conjunctions isomorphism means that their arguments may be renamed in such a way that they coincide up to the order of literals.

Note that both objective variables and objective constants may be used for arguments of elementary conjunctions P and Q. The notion of elementary conjunctions isomorphism differs from the one of the equivalence of formulas, because the elementary conjunctions may have essentially different arguments. Essentially, there are such permutations of these formulas arguments that they define the same relation between their arguments and this relation is defined by the formula R with variables as its arguments.

3 Partial sequence

The problem of checking if the formula $A(\overline{x})$ or some its sub-formula $\widetilde{A}(\overline{y})$ is a consequence of the set of formulas $S(\omega)$ is under consideration in [4]. Here the list of arguments $\overline{y}$ is a sub-list of arguments $\overline{x}$.

Every sub-formula $\widetilde{A}(\overline{y})$ of the formula $A(\overline{x})$ is called its **fragment**.

Let a and $\widetilde{a}$ be the numbers of atomic formulas in $A(\overline{x})$ and $\widetilde{A}(\overline{y})$, respectively, m and $\widetilde{m}$ be the numbers

of objective variables in $\overline{x}$ and $\overline{y}$, respectively.

Numbers q and r are calculated by the formulas $q = \frac{\widetilde{a}}{a}$, $r = \frac{\widetilde{m}}{m}$ and characterize the degree of coincidence between $A(\overline{x})$ and $\widetilde{A}(\overline{y})$. For every $A(\overline{x})$ and its fragment $\widetilde{A}(\overline{y})$ it is true that $0 < q \leq 1$, $0 < r \leq 1$. Besides, $q = r = 1$ if and only if $\widetilde{A}(\overline{y})$ coincides with $A(\overline{x})$.

Under these notations, the formula $\widetilde{A}(\overline{y})$ will be called a (q, r)**-fragment of the formula** $A(\overline{x})$.

If $S(\omega) \Rightarrow \exists \overline{x}_{\neq}\ A(\overline{x})$ is not valid but for some (q, r)-fragment $\widetilde{A}(\overline{y})$ $(q \neq 1)$ of $A(\overline{x})$ the sequent $S(\omega) \Rightarrow \exists \overline{y}_{\neq}\ \widetilde{A}(\overline{y})$ is true, we will say that $S(\omega) \Rightarrow_P \exists \overline{x}_{\neq}\ \widetilde{A}(\overline{x})$ is a **partial** (q, r)**-sequent**.

As the checking whether $S(\omega) \Rightarrow \exists \overline{y}_{\neq}\ \widetilde{A}(\overline{y})$ may be done by some constructive method (for example, by exhaustive search or by deduction in a sequential predicate calculus) then such values $\overline{\tau}$ $(\tau \subseteq \omega)$ for the list of variables $\overline{y}$ that $S(\omega) \Rightarrow \widetilde{A}(\overline{\tau})$ will be found.

Definition 2. *Conjunction of literals from $A(\overline{x})$ which are not in $\widetilde{A}(\overline{y})$ is called a* **complement** *of $\widetilde{A}(\overline{y})$ up to $A(\overline{x})$.*

A complement of $\widetilde{A}(\overline{y})$ up to $A(\overline{x})$ will be denoted by $C_{A(\overline{x})}\widetilde{A}(\overline{y})$.

Definition 3. *A (q, r)-fragment $\widetilde{A}(\overline{y})$ of the formula $A(\overline{x})$ is called* **contradictory** *to the description $S(\omega)$ on the list of constants $\overline{\tau}$ if $S(\omega)$ and $C_{[A(\overline{x})]^{\overline{y}}_{\overline{\tau}}}\widetilde{A}(\overline{\tau})$ lead to the contradiction, i.e., $S(\omega) \Rightarrow \neg C_{[A(\overline{x})]^{\overline{y}}_{\overline{\tau}}}\widetilde{A}(\overline{\tau})$.*

Here the denotation $[A(\overline{x})]^{\overline{y}}_{\overline{\tau}}$ is used for the result of substitution of the constants from the list $\overline{\tau}$ instead of the corresponding variables from the list $\overline{y}$.

4 Level description of classes

To decrease the number of the algorithm run steps while solving the problems (1), (2), (3) and (4) a level description of classes was suggested in [3]. In fact, such a description is a hierarchical one and is based on extraction from the class description some sub-formulas which are isomorphic to each other and define generalized characteristics of objects from one class [6, 9].

It may be done by means of extraction of formulas $P_i^1(\overline{y}_i^1)$ which are isimorphic to "frequently" appeared sub-formulas of $A_k(\overline{x}_k)$ $(k = 1, \ldots, K)$ with "small complexity". At the same, time a system of equivalences of the form $p_i{}^1(y_i^1) \leftrightarrow P_i^1(\overline{y}_i^1)$ is written down. Here $p_i{}^1$ are new predicates of the 1st level and y_i^1 are new 1st level variables for lists of initial variables.

Denote formulas received from $A_k(\overline{x})$ by means of substitution of $p_i^1(x^1_{i\ j})$ [3] instead of all occurrences of sub-formulas isomorphic to $P_i^1(\overline{y}_i^1)$, by $A_k^1(\overline{x}_k^1)$. Here $\overline{x}_k^1$ is the list of all variables of $A_k^1(\overline{x}^1)$ which includes both some (may be all) initial variables and the 1st level variables in the formula $A_k^1(\overline{x}^1)$. These formulas may be regarded as class descriptions in the terms of initial (0-th) and 1st level predicates.

The 1st level description $S^1(\omega)$ of the object ω is a union of $S(\omega)$ and the set of all constant atomic formulas $p_i^1(\tau_{ij}^1)$ such that the 1st level constant τ_{ij}^1 is the name of a list of initial objects $\overline{\tau}_{ij}^1$ and $p_i^1(\tau_{ij}^1) \Leftrightarrow P_i^1(\overline{\tau}_{ij}^1)$.

The process of extraction of sub-formulas $P_i^l(\overline{y}_i^l)$ which are isomorphic to "frequently" appeared sub-formulas of $A_k^{l-1}(\overline{x}_k^{l-1})$ with "small complexity", may be repeated for $l = 1, \ldots, L$.

A level description of classes has the form

$$\begin{cases} p_1^1(x_1^1) & \Leftrightarrow & P_1^1(\overline{y_1^1}) \\ & \vdots & \\ p_{n_1}^1(x_{n_1}^1) & \Leftrightarrow & P_{n_1}^1(\overline{y_{n_1}^1}) \\ & \vdots & \\ p_i^l(x_i^l) & \Leftrightarrow & P_i^l(\overline{y_i^l}) \\ & \vdots & \\ p_{n_L}^L(x_{n_L}^L) & \Leftrightarrow & P_{n_L}^L(\overline{y_{n_L}^L}) \\ & A_k^L(\overline{x}_k^L) & \end{cases} .$$

The solving of the described recognition problems may be reduced to the sequential check for $l = 1, \ldots, L$ of the following operation.

- For every $i = 1, \ldots, n_l$ check whether $S^{l-1}(\omega) \Rightarrow \exists \overline{x}_{i\neq}^l\ P_i^l(\overline{x}_i^l)$ and find all such l-level objects $\overline{\tau}_{ij}^1$ that $S^{l-1}(\omega) \Rightarrow P_i^l(\overline{\tau}_{ij}^l)$.

Note that for every l the sequence in the form (4) is checked. That's why the term formula of "small complexity" may be precised. For an exhaustive search algorithm it means a small number of variables. For deduction in a sequential predicate calculus it means a small number of atomic formulas.

An algorithm of the extraction of the maximal formula, which is isomorphic to sub-formulas of two given elementary conjunction and construction of a level description of class, is in [8].

[3] Index j is between 1 and the number of occurrences of sub-formulas isomorphic to $P_i^1(\overline{y}_i^1)$.

5 Logic-predicate networks

Logic-predicate network contains two blocks: a training block and a recognizing one [7].

During the initial construction of the network a training set (TS) is given to form a recognizing block. TS contains descriptions of the objects with pointing out the class number. Different constants in the object description are changed by different variables and the sign & is put between literals of the object description. Then an algorithm of the level description of classes corresponding to the TS is used.

Formulas $P_i^1(\overline{y}_i^1)$ $(i = 1,\dots n_l,\ l = 1,\dots,L)$ are the matter of l-level cells of the recognizing block. The last level contains formulas $A_k^L(\overline{x}_k^L)$.

The recognizing block runs according to the algorithm of level recognition.

If during the use of the network there is a wrong recognition then it is possible to retrain the network by means of adding the description of wrong recognized object to the first level of the training block. Recognizing block may be reconstructed with changing the number of levels and the number of cells in some levels. The scheme of recognizing block of logic-predicate network is presented in figure1.

As it can be seen from this scheme, a logic-predicate network is an oriented graph in every node of which similar sequences in the form $S^l(\omega) \Rightarrow \exists\overline{x}_{i\neq}^l P_i^l(\overline{x}_i^l)$ are checked and parts of the recognizable object satisfying the formula are found.

Unfortunately an object that is even slightly different from an object in TS (their descriptions are not isomorphic) would not be recognized. To recognize such an object, it is necessary to retrain the network.

6 Fuzzy recognition by logic-predicate networks

In this paper it is suggested to change the content of the network cells by replacing the checking of $S^{l-1}(\omega) \Rightarrow \exists\overline{x}_{i\neq}^l P_i^l(\overline{x}_i^l)$ in the ith cell of the lth level with the partial sequence checking $S^{l-1}(\omega) \Rightarrow_P \exists\overline{x}_{i\neq}^l P_i^l(\overline{x}_i^l)$.

While partial sequence checking, all lists of constants $\overline{\tau}_{i\ j}^l$ and maximal not contradictory on $\overline{\tau}_{i\ j}^l$ with $S^{l-1}(\omega)$ sub-formula $\widetilde{P}_{i\ j}^l(\overline{x}_{i\ j}^l)$ of the formula $P_i^l(\overline{x}_i^l)$ are found. Parameters $q_{i\ j}^l$ and $r_{i\ j}^l$ are calculated with the use of the full form of formulas $P_i^l(\overline{x}_i^l)$ and $\widetilde{P}_i^l(\overline{y}_i^l)$, i.e., with the replacement in them of each atomic formula of the levels l' $(l' < l)$ by the defining elementary conjunction.

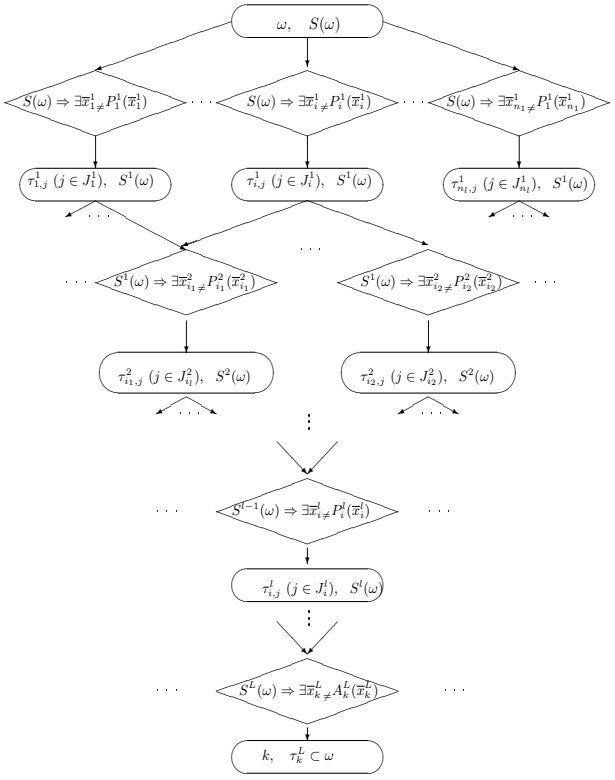

Figure 1: Scheme of logic-predicate network recognizing block

Except this, a "degree of certainty" $cert_i^l$ that the recognition would be valid is calculated in every cell. Denotation pre_i^l will be used for the number of the cell preceding the ith cell of the lth level while the current graph traversal. For example, $pre_{i_1}^2$ in the fig. 1 equals 1 while one traversal and equals i while another traversal.

Initially, $cert_1^0 = 1$, $pre_i^1 = 0$ $(i = 1,\dots,n_1)$. While first visit of the ith cell of the lth level $cert_i^l := \min\{cert_{pre_i^l}^{1-1}, q_i^l\}$. This corresponds to conjunction of degrees of certainty while successive passage along one branch of network traversal. While next visit of the ith cell of the lth level $cert_i^l := \max\{cert_i^l, \min\{cert_{pre_i^l}^{1-1}, q_i^l\}\}$. This corresponds to disjunction of degrees of certainty while parallel passage along different branches of network traversal.

In the last level of the network numbers of classes and objects will be received. For every such an object a degree of certainty that it is a r_i^Lth part of an object of this class will be calculated.

To increase the velocity of the network run it may be offered that the graph traversal is a combination of wide graph traversal and depth traversal. In such a

case for every level when queuing nodes we will sort the parameters q_i^l in such a way that $q_{i_1}^l \geq q_{i_2}^l \geq \ldots \geq q_{i_{n_l}}^l$. After that insert all these nodes (regardless of whether they are yet in the queue or not) into the queue so that the whole queue consists of a non-increasing sequence.

If the object can be recognized exactly (i.e., there is a branch of the traversal, in each node of which $cert_i^l = 1$), then the graph will be traversed in depth along a single branch.

In addition, it is possible to set the value of the parameter α and remove from consideration cells where $cert_i^l < \alpha$.

7 Conclusion

A modification of logic-predicate recognition network is suggested in the paper. With the use of such a network, it is possible not to accurately determine the class to which the recognized object belongs, but also to calculate the "degree of certainty", that the recognition is correct.

As in the original version of the logical-predicate recognition network, this modification has the possibility of its further retraining if the object was correctly recognized even with a small degree of certainty.

In the case of exact recognition of the object, the computational complexity of the modified network run is almost the same as when using the original network. More accurate estimates of computational complexity are planned for further work.

References

[1] C.M. Bishop, Neural Networks for Pattern Recognition., Oxford University Press, Inc. New York, NY, 1995.

[2] M.R. Garey and D.S. Johnson, Computers and Intractability: a Guide to the Theory of NP-completeness, Freeman, New York, 1979.

[3] T.M. Kosovskaya, Level descriptions of classes for decreasing of step number of solving of a pattern recognition problem described by predicate calculus formulas, Vestnik of Saint-Petersburg University. Series 10. Applied mathematics. Computer science. Control processes, issue 1 (2008) 64 – 72. (In Russian)

[4] T.M. Kossovskaya, Partial deduction of predicate formula as an instrument for recognition of an object with incomplete description, Vestnik of Saint-Petersburg University. Series 10. Applied mathematics. Computer science. Control processes, issue 1 (2009) 74 – 84. (In Russian)

[5] T.M. Kosovskaya, Some artificial intelligence problems permitting formalization by means of predicate calculus language and upper bounds of their solution steps, SPIIRAS Proceedings, **14** (2010) 58 – 75. (In Russian)

[6] T.M. Kosovskaya, An approach to the construction of a level description of classes by means of a predicate calculus language, SPIIRAS Proceedings, **3(34)**(2010) 204 – 217. (In Russian)

[7] T.M. Kosovskaya, Self-training Network with the Sells Implementing Predicate Formulas, SPIIRAS Proceedings, **6(43)**(2015) 94 – 113. (In Russian)

[8] T.M. Kosovskaya and D.A. Petrov, Extraction of a maximal common sub-formula of predicate formulas for the solving of some Artificial Intelligence problems, Vestnik of Saint-Petersburg University. Series 10. Applied mathematics. Computer science. Control processes, issue 3(2017) 250 – 263. (In Russian)

[9] T.M. 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 (2018) 1 – 20.

https://www.intechopen.com/books/intelligent-system/predicate-calculus-as-a-tool-for-ai-problems-solution-algorithms-and-their-complexity

[10] R.E. Neapolitan, Learning Bayesian Networks, Pearson Publ, 2003.

[11] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Third edition, Prentice Hall Press Upper Saddle River, NJ, 2009.

[12] A. Tulupiev, C. Nikolenko and A. Sirotkin, Bayesian Networks: logic-probabilistic approach, Nauka, 2006. (in Russian)