ISSN 0002-306X. Proc. of the RA NAS and NPUA Ser. of tech. sc. 2019. V. LXXII, N2.

UDC 004.8

COMPUTER SCIENCE
AND INFORMATICS

N.P. HAKOBYAN

A SYSTEM FOR TRANSFORMING IMAGES TO SYMBOLIC PRESENTATION FOR COMBINATORIAL DEFENSE AND COMPETITION PROBLEMS

We aim to develop tools for regular transformation of combinatorial defense and competition problems' situations to their symbolic presentation by machine learning solutions. As it is proven, that RGT class problems are reducible to each other, in this paper we demonstrate a developed ANN model to detect the chess board from an image and classify the chess pieces. Two models were developed – a simple one which provides high accuracy as some complex models worldwide, and the second approach (based on a new method), which correctly fits for other RGT problems with a bit lower accuracy.

Keywords: neural networks, image classification, object detection, systemic classifications, algorithms.

1. Introduction

1.1. Background of the research

Cognitive Algorithms and Models research directions include the ones in Artificial Intelligence and aim to develop constructively regularized models of human approaches in solving combinatorial problems.

There are different lines of researches in this area, e.g. machine learning solutions, such as neural networks, that concentrate on modeling of biological nature of the human brain.

Following the line of Cognitive Algorithms and Models research directions, we concentrate on the development and applications of cognitive functions to a class of combinatorial problems defined as problems where spaces of solutions are Reproducible Game Trees (RGT) [1-3].

1.2. RGT class

RGT class includes important problems like computer networks intrusion protection, optimal management and marketing strategy elaboration in competitive environments, defense of military units from a variety types of attacks, communication problems, certain types of teaching, chess and chess-like games [2].

In the continuous researches of our team, a class of problems is defined as a class of unsolved combinatorial problems [3].

The class named RGT is a subclass of Optimal Strategy Provision (OSP) problems. The RGT problems meet the following requirements:

- there are (a) interacting actors (players, competitors, etc), performing (b) identified types of actions in the (c) specified types of situations;
- there are identified utilities, goals for each actor;
- actions for each actor are defined.

Actors perform their actions in specified periods of times and do affect situations by actions in time t by transforming them to new situations in time t+1 trying to achieve the best utilities on that situations (goals) by regularities defining these actions.

1.2.1. Achievements in RGT and some open questions

There are certain achievements for the RGT class, some of which are listed below:

The proposed in [4,5] theory of mental doings provides ways for constructive and adequate models of various cognitive functions, e.g. classification, explanation, etc. We have implemented a knowledge-based expert system, RGT Solver18, which is able to utilize those cognitive functions.

The developed software can examine the developed models for systemic classifiers of RGT problems: algorithms and structures have been developed to provide an adequate description of systemic classifiers and to ensure their correspondence [6], and their adequacy were demonstrated by experiments [7].

RGT class combinatorial problems are reducible to each other [8].

The results of the work are applicable to the actual problems of real-time detection of means of attack and protection of the enemy with the help of autonomous drones and making the best decisions of counteraction (attack, retreat, investigation) in such situations. In [9], the defense of navy from air threats is described as a RGT class problem and certain solutions are provided.

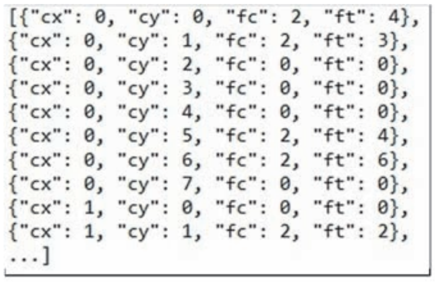
The natural situations of different problems are presented differently, while the transformation of natural situations (images) into software-presented ones (symbolically) is another widely discussed topic of the study with frequently provided solutions of machine learning [10, 11].

Particularly, in [11], the author introduces a new method of object detection, which can be useful for some RGT problems situations' processing and ensures the effectiveness and fast-working process. It also detects the hidden objects, which cannot be detected adequately by a human without any tools. However, the solution is provided for sequential frames processing and not for single images.

We consider the transformation of a situation to Solver as a computer vision problem, particularly, image classification and object detection. Since Neural networks are the leading method of solving such problems all over the world, we have also chosen them as a tool.

Currently, the first layer of Solvers is a symbolical input (Fig. 1), which is done by experts/programmers.

The natural presentation of situations are different (Fig. 2) and, currently, there is no way to pass the image presentations of situations to the Solver in a regular way. Thus, in the current work we aim to develop an interface for regular transformation of natural presentations of situations into Solver environment. The problem can be divided into two subtasks:



```
[{"cx": 0, "cy": 0, "fc": 2, "ft": 4},
{"cx": 0, "cy": 1, "fc": 2, "ft": 3},
{"cx": 0, "cy": 2, "fc": 0, "ft": 0},
{"cx": 0, "cy": 3, "fc": 0, "ft": 0},
{"cx": 0, "cy": 4, "fc": 0, "ft": 0},
{"cx": 0, "cy": 5, "fc": 2, "ft": 4},
{"cx": 0, "cy": 6, "fc": 2, "ft": 6},
{"cx": 0, "cy": 7, "fc": 0, "ft": 0},
{"cx": 1, "cy": 0, "fc": 0, "ft": 0},
{"cx": 1, "cy": 1, "fc": 2, "ft": 2},
...]
```

Fig. 1. Symbolic presentation in the Solver

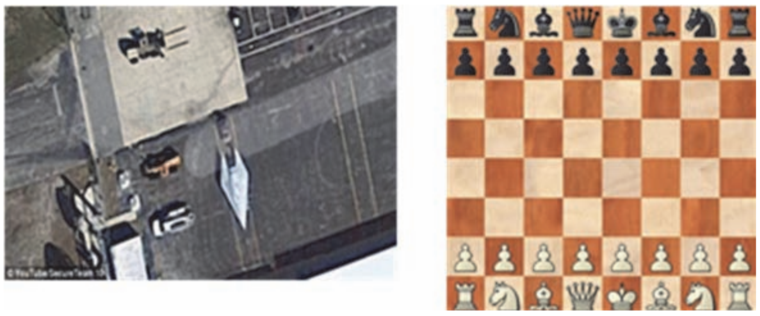


Fig. 2. Natural Presentations of Situations: Battles (Left) and Chess (right)

a. Detecting the situations from the given image.

b. Modifying the situation to acceptable for the Solver form.

1.2.2. Battle Field as RGT class problem

The Battle Field can be considered as a RGT problem by the following interpretation (Fig. 3):

1. The battling sides can be considered as interacting actors;

2. Military units' movements, attacks can be considered as actions;

3. The battle field area including the military units can be considered as situations;

4. Different situations can be considered as goals: capture objects, destroy enemy units, push frontline.

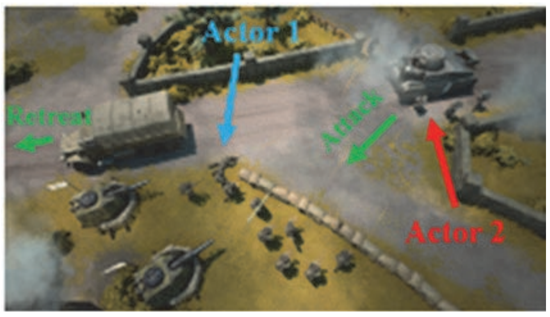


Fig. 3. Actors and Actions in Battle Field Problem

1.3. Related Works

During our research several works to detect the chess board and do piece classification were considered. In most researches (3D objects) the board, detection is done via image processing and not machine learning by using the already developed model from opencv. (canny edge detection and Hough line detection) [12].

In [13], the author describes the developed model for 3D objects/images, using the mentioned opencv model for board detection and Caffe deep learning framework and pre-trained AlexNet [14] and achieved pretty high 99% accuracy. The shortcomings of the model are a few: a) the model is too dependent on certain construction of the board and pieces, i.e. it works badly for the pieces and the board of other construction than the training set does.

The Chessify project, launched by Fimetech LLC ([15]) in 2016 provides a solution for chess board detection and piece classification for 2D objects with pretty high accuracy. However, Chessify is the best among similar solutions worldwide by universalization, it still supports only 2D detections and is not open-source.

In [16] the model described by the author for the chess board detection avoids using the opencv module but includes the manual selection of the four corners of the board in the specified order, which we also aim to automatize since in other than chess situations it might be hard to specify boundaries of the situation manually. However, the model uses SVM for training and is simpler than other models, the accuracy of the pieces' classification is much lower (~85%), and it is still dependent on certain shapes of figures (training dataset).

All the described models and others researched have several disadvantages that we aim to avoid, if possible, while building our own model – a) dependence over certain shapes of board and pieces (the most common issue); b) complex models; c) low accuracy.

1.4. Current work

Currently, the input layer in the RGT Solver is provided by experts/ programmers in a symbolical way.

We aim to develop a tool for regular passing of natural (image) presentations of RGT situations to symbolical ones in the RGT Solver. For this purpose, we are using ANN.

As it is proven, the RGT class problems are reducible to each other, we are providing experiments for chess, so the current work concentrates on the chess situations and pieces.

The existing models offer a classification for certain types of pieces – training of NN was done on the exact board and pieces. Its achievement is 99 % by the test (same types), but if we change the figure shapes somehow (take another board and other pieces with different shapes) the result will be lower. We aim to enhance the classification to be universal for detection of different shapes of figures.

For the users' convenience, we have developed models both for 2D and 3D images. For experimenting we concentrate more on 2D objects.

We discuss the parallels and the possibility of transmission of the achieved results of chess to the battle field problem in this paper as well.

We provide two algorithms for our purpose – each of them will be described in separate sections.

2. Simple Algorithm

The task is as follows: Given the image of the chess board, it is necessary to classify each field as an empty one or a certain figure depending on its color and type (black rook, white pawn etc).

In other words, this task can be stated as follows: to classify nuclear classifiers from the situation, which are attributes of pieces for chess: color, type and coordinates.

2.1. Chess Piece Classification

2.1.1. 2D images

2.1.1.1. Dataset

A dataset of around 300 images of chess boards with existing figures on them was collected. A Python script was written to split the board into 64 equal squares (8 rows, 8 columns). The pieces constructions are of various types which insures universalization of the model. Each of the 64 received images of every picture includes either an empty square or a figure with certain color and figure type, which was annotated. The dataset was split randomly into training and testing sets by a 3:1 ratio. There are overall 13 classes numerated from 0 to 12, where 0 corresponds to an empty field, 1 to 6 is for white pawn, bishop, knight, rook, queen and king and from 7 to 12 for black pieces with the same order. Some samples from the dataset are shown in Fig. 4.

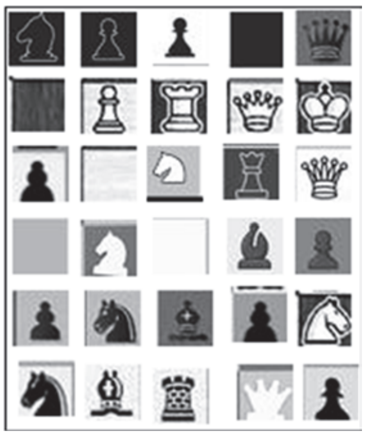


Fig. 4. Samples from Dataset of Chess Pieces

2.1.1.2. Learning

Keras was chosen as the neural network library for our model, which was trained to classify the pieces on images on the described dataset. VGG-like ([13]) convolutional network of the following construction is selected:

- 2 convolutional layers with 32 neurons and kernel size of (3,3),
- Max-Pooling layer with a pooling size (2,2),
- 2 convolutional layers with 64 neurons and kernel size of (3,3),
- Max-Pooling layer with a pooling size (2,2),
- Flattening the 2D arrays for fully connected layers,
- 3 Dense layers with 256, 128, 64 neurons correspondingly and with RELU activation,
- The last layer is dense layer with 13 classes (for each of classes of our classification) and Softmax activation.

2.1.1.3. Results

The accuracy of 97.3 % on the test dataset of piece classification was achieved. The corresponding confusion matrix is provided in Fig. 5.

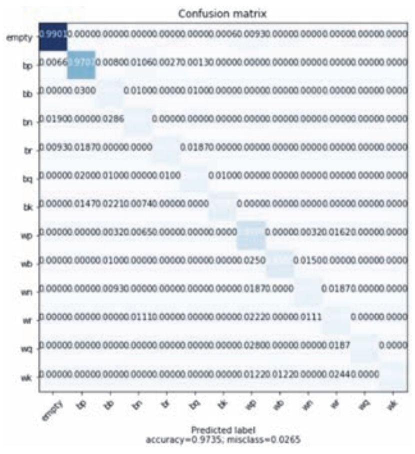


Fig. 5. Confusion Matrix of Piece Classification

2.1.2. 3D images

[13] was taken as the base for this work. The dataset was collected as combination of datasets provided in [13], [16] and manually collected dataset of various constructions, which ensures some universalization of the model. As long as all our models are built in Keras, and there is no built-in model for AlexNet there, we choose the VGG-16 as our model. Accuracy of 94.2 % is achieved.

3. Enhanced Algorithm

The algorithm includes several steps

3.1. Detecting Pieces

The dataset of chessboard images including the chess pieces and empty fields was collected. All chess figures were marked and labeled in each of these images using LabelImg [18]. We have built ANN based on the collected dataset to detect the chess pieces from the board.

3.2. Coordinate Comparison

After detecting the pieces, we have a set of pieces with coordinates (x1; x2; y1; y2).

For farther simplicity, let’s denote the following:

$$F - set\ of\ received\ figures$$
$$X, Y - set\ of\ received\ coordinates\ (both\ from\ R^2\ dimension)$$
$$coordinates(x_i'; y_i'; x_i''; y_i'')\ correspond\ to\ the\ figure\ f_i$$
$$A_x = avg(x'' - x'), for(x'; x'') \in X - the\ mean\ side\ of\ field\ (horizontal)$$
$$A_y = avg(y'' - y'), for(y'; y'') \in Y - the\ mean\ side\ of\ field\ (vertical)$$

The next step is to understand how many rows or columns are between certain two figures. As long as all the fields' dimensions are nearly equal (the more the angle of shooting differs from 90^o, the less equal they are), we use the following equation to determine the number of the rows between figures f' and f'' with coordinates [y1', y2'] and [y1'', y2'']:

$$r \sim \frac{(y_2' - y_1') + (y_2'' - y_1'')}{2 \text{ x } A_y}, \qquad (1)$$

where r is the number of the rows between the figures. In the future we will research how this equation should change if the angle of shooting is higher or lower than 90^o. Here we just round the received number to the nearest integer (it can also be negative, negative rows mean movement to the left and negative columns mean movement down).

The same is done for columns and the equation is analogic.

3.3. Creating Initial Board

Now it is time to merge the received figures together. The algorithm works as follows: one of the pieces is taken as an initial point with the row index and the column index equal to 0; on each iteration, the program chooses a not yet considered figure and counts the number of rows and columns by equation (1) and saves the results as a tuple (f, r, c), where f is the figure type (black bishop, white rook etc), r and c are the indexes of row and column correspondingly (can be negative).

After all iterations, the program gets the minimum number of all column indexes, if it is negative, it sums all the received tuples' third element by the absolute value of the minimum number. The same thing is being done for rows.

At the end, we receive coordinates for each figure.

We fill the missing values with empty fields – for example, if we have some figures on coordinates (0;0) and (0;3), but there is nothing in (0;1) and (0;2), we fill them with empty fields.

3.4. Extending Board by missing lines/rows

If the initial board had an empty corner row(s) or column(s) they would be missing after these steps and we would have a matrix with dimensions 7x8 (8x7) (if only one row (column) was empty). The task is to find on which of 2 sides the empty row must be added.

Actually, this task is local for chess and probably would not appear in other RGT class problems. For instance, in the Battle Field problem, after detecting all military units, all ‘fields’ or ‘areas’ around (maybe with some limited range depending on zone where it is possible to hold fights) can be considered as empty. Of course, this brings about another task, to detect, for example, a field, a flatland, or which parts of the area are plateau, which can affect the possible movements there, but this is also a local problem and would be considered in future in more detailed researches on the Battle Field problem.

We take all empty fields we have in our received board and compare the images (currently we use histogram comparison, but this approach has to be improved and will be researched in future) of those fields to each of 8 parts of a new line from top. If all the 8 parts match (more than the parts of the other line) with some empty fields in our board, then we consider the line as searched. We do the same for each side and we make as many iterations as many lines are missing.

4. Application in the RGT Solver

After both modules for board detection and piece classification are ready, the already classified figures and empty fields are processed to JSON format as it is described in the picture below. The Solver receives a list of 64 JSONs as an input (Fig. 1) each of which refers to a certain field and contains nuclear classifiers’ values. For example, the first raw of this image corresponds to the field with coordinates (0, 0), which is a8 on chess board, figure color on that field is black (2) and figure type is rook (4). Finally, the chess situation in the usual for the Solver format is achieved and systemic classifications are processed.

Table 1

Comparison of the Results

Criterion/Method	**Simple Algorithm**	**Enhanced Algorithm**	Chessify	Chess ID	ChessVision
Accuracy	97.3%	95.1%	Unknown (high)	99%	85%
Universalization	Yes	Yes	Yes	No	No
Open-Source	Yes	Yes	No	Yes	Yes
Costly Training	No	Yes	Unknown	Yes	No
Used Model or Method	VGG-like ANN	Mobilenet SSD	Unknown	AlexNet	SVM
Automate Board Detection	Yes (openCV)	Yes	Yes	Yes (openCV)	No (needs marking)

5. Parallels with Battle Field Problem

5.1. Classifying Units

For the Battle Field problem, tanks, rockets etc can be considered as units. Military units in the given image should be classified similar to the units in chess.

5.2. Difficulties

Some difficulties appear at transforming the chess advances to the battle field:

1. Quality data – if the chess piece constructions are the same all over the world, the weapons are different in different countries. Anyway, the most popular types are of the same construction (e.g. tanks)

2. Classifying the actor/side – for the chess it is simple – the actor can be classified just by the piece's color. For the battle field, it is harder because battling sides' military units of the same type difference is sometimes even harder to detect by a human. This is still possible to realize, it is just harder than the corresponding task for chess.

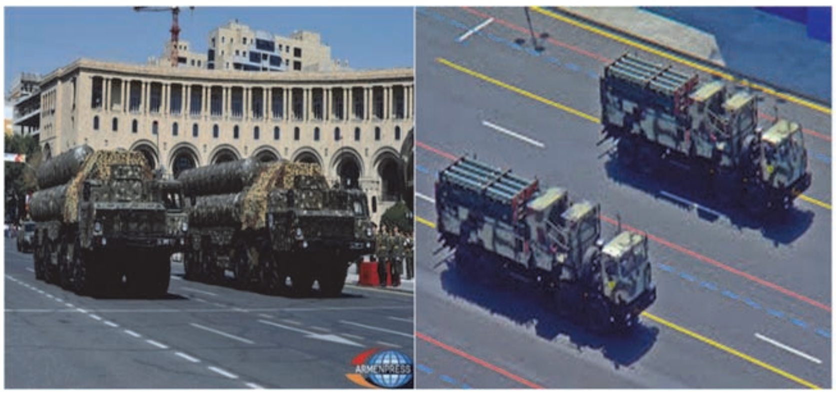


Fig. 6. Military Units of Different Countries: Armenia (left) and Azerbaijan (right)

3. Angle – Angle of unit in an image and camera was not much an important task for chess, as long as it is not hard to change the angle for the photographer, while for the battlefield it is sometimes possible to shoot only from a limited number of places/angles. These also makes the minimal required quality of the dataset higher.

Conclusion

Methods for the situation's natural presentation transformation to Solver's symbolic presentation are proposed. Algorithms use Neural Networks for classification/detection of units and are described for certain RGT class problem – chess. The first algorithm includes classification of chess pieces, the second one – detecting the chess pieces from an image and detecting board by them. The discussed models were integrated with Solver 18. The possible applications of solutions for battle field problems are described and some difficulties over battle field interpretation by chess solutions are listed. Implementation of the model for Battle Field problem, including dataset collection, researching ways for solution of described difficulties and ANN training are considered as the future steps.

References

1. **Pogossian E.** Adaptation of Combinatorial Algorithms. - Yerevan: Academy of Sciences of Armenia, 1983. - 293 p.
2. **Pogossian E.** On Modeling Cognition // Computer Science and Information Technologies (CSIT11). - Yerevan, Sept.26-30, 2011.
3. **Grigoryan S.** Research and development of algorithms and programs of knowledge acquisition and their effective application to resistance problems: PhD. - Yerevan, Armenia, 2016. - 111 p.
4. **Pogossian E.** Constructing adequate mental models // Transactions of IIAP NAS RA, Mathematical Problems of Computer Sciences. – 2018. - Vol. 50. - P. 35-51.
5. **Pogossian E., Grigoryan S., Hakobyan N.** On Systemic Classifications and Machine Learning // Computer Science and Information Technologies. – 2017. - P. 102-108.
6. **Grigoryan S., Hakobyan N. and Vrtanesyan H.** Object–oriented modeling of matching to systemic classifiers // Transactions of IIAP NAS RA: Mathematical Problems of Computer Sciences.- 2018. - Vol. 48. - P. 115-121.
7. **Grigoryan S., Hakobyan N.** Experimenting with acquisition of and matching to Systemic Classifiers // Transactions of IIAP NAS RA: Mathematical Problems of Computer Sciences. – 2018. - Vol. 50. -P. 96-103.
8. **Pogossian E.** Combinatorial Game Models For Security Systems, in NATO ARW on “Security and Embedded Systems”.- Porto Rio, Patras, Greece, 2005.
9. **Pogossian E.** Effectiveness Enhancing Knowledge Based Strategies for SSRGT Class of Defense Problems NATO ASI 2011 Prediction and Recognition of Piracy Efforts Using Collaborative Human-Centric Information Systems.- Salamanca, Spain, 2011.
10. **He, K., Ren, S., Sun, J., & Zhang, X**. Deep Residual Learning for Image Recognition.- 2016.- CoRR, abs/1512.03385.
11. **Simonyan R.**, Development of object detection, classification and positioning system: PhD. - Yerevan, Armenia, 2018. - 115 p.
12. https://docs.opencv.org/2.4.13.7/doc/tutorials/calib3d/camera_calibration_square_chess/camera_calibration_square_chess.html
13. **Yang D.,** Building Chess Id, https://medium.com/@daylenyang/building-chess-id-99afa57326cd, 2016.
14. https://en.wikipedia.org/wiki/AlexNet
15. https://chessify.me
16. **Ding J.**, ChessVision: Chess Board and Piece Recognition, 2016.
17. **Simonyan K., Zisserman A.**, Very Deep Convolutional Networks for Large-Scale Image Recognition - Computer Vision and Pattern Recognition, 2015.

18. https://github.com/tzutalin/labelImg

Russian-Armenian (Slavonic) University. The material is received 15.04.2019.

Ն.Պ. ՀԱԿՈԲՅԱՆ

ՊԱՏԿԵՐՆԵՐԸ ՍԻՄՎՈԼԻԿ ՆԵՐԿԱՅԱՑՄԱՆ ՓՈԽԱՐԿԵԼՈՒ ՀԱՄԱԿԱՐԳ ՊԱՇՏՊԱՆՈՒԹՅԱՆ ԵՎ ՄՐՑԱԿՑԱՅԻՆ ԿՈՄԲԻՆԱՏՈՐ ԽՆԴԻՐՆԵՐԻ ՀԱՄԱՐ

Առաջարկվել են իրավիճակի բնական ներկայացումը Solver ծրագրային ապահովման սիմվոլիկ ներկայացմամբ փոխակերպելու մեթոդներ: Ալգորիթմներում, որոնք նկարագրվում են RGT դասի որոշակի խնդրի՝ շախմատի համար, կիրառվում է նեյրոնային ցանցերով ուսուցում՝ օբյեկտների ճանաչման և հայտնաբերման համար: Առաջին ալգորիթմը ներառում է շախմատային խաղաքարերի դասակարգում, երկրորդը՝ պատկերից շախմատային խաղաքարերի, իսկ վերջիններիս միջոցով՝ տախտակի ճանաչում: Վերոնշյալ մոդելները ինտեգրվել են Solver18-ին:

Նկարագրվում է ստացված լուծումների հնարավոր կիրառությունը ռազմական որոշակի խնդիրներում, և քննարկվում են այդ խնդիրների մեկնաբանման հետ կապված որոշ դժվարություններ: Որպես հետագա քայլ դիտարկվում է մոդելի իրականացումը ռազմական խնդիրների համար, այդ թվում՝ տվյալների հավաքագրում, նկարագրված դժվարությունները հաղթահարելու ուղիների հայտնաբերում և նեյրոնային ցանցերի միջոցով ռազմական միավորների ուսուցում:

Առանցքային բառեր. նեյրոնային ցանցեր, պատկերի ճանաչում, օբյեկտի հայտնաբերում, սիստեմիկ դասակարգիչներ, ալգորիթմներ:

Н.П. АКОПЯН

СИСТЕМА ПЕРЕВОДА ИЗОБРАЖЕНИЙ В СИМВОЛЬНОЕ ПРЕДСТАВЛЕНИЕ ДЛЯ КОМБИНАТОРНЫХ ЗАДАЧ ЗАЩИТЫ И КОНКУРЕНЦИИ

Предложены методы преобразования естественных представлений ситуаций в символьное представление Solver-а. Алгоритмы используют нейронные сети для распознавания/обнаружения объектов и описаны для определенной задачи класса RGT - шахмат. Первый алгоритм включает распознавание шахматных фигур, второй - обнаружение шахматных фигур по изображению и нахождение доски по ним. Обсуждаемые модели интегрированы с Solver18.

Описаны возможности применения полученных решений в военных задачах (battlefield problem) и перечислены некоторые трудности, связанные с интерпретацией данной задачи полученными решениями. В качестве следующих шагов рассматривается реализация модели для проблемы Battle Field, включая сбор данных, поиск путей решения описанных трудностей и обучение посредством нейронных сетей.

Ключевые слова: нейронные сети, распознавание изображения, обнаружение объектов, системные классификаторы, алгоритмы.