INFORMATYKA
CZASOPISMO TECHNICZNE
TECHNICAL TRANSACTION
COMPUTER SCIENCE AND INFORMATION SYSTEMS
WYDAWNICTWO POLITECHNIKI KRAKOWSKIEJ
1-I/2011
ZESZYT 9
ROK 108
ISSUE 9
YEAR 108

BARBARA ŁUKAWSKA*

DYNAMIC CLASSIFICATION: A NOVEL APPROACH TO SELECTION OF COMPLEX OBJECTS

DYNAMICZNA KLASYFIKACJA: NOWE PODEJŚCIE DO PROBLEMU SELEKCJI SKOMPLIKOWANYCH OBIEKTÓW

Abstract

The problem of selecting the best objects may be solved with the help of classification tools. There are many software tools helpful in data exploration, and in the classification, but all of them have some restrictions. Selection algorithms used in these tools are of no use for data burdened with errors and noise, or if the information should be dynamically processed during the experiment. The problem is analyzed using selection of candidates for mobile robot (mobot) operators as an example. The paper includes comparison of basic classification methods (classification trees, rough sets and fuzzy sets), too. Moreover, a new software tool, effective in candidate selection, is presented. The tool is especially useful for candidates for difficult tasks and jobs requiring special predispositions. Presented classifier works with data burdened with noise, and also if the data is gathering dynamically, after the classification has started. In may be used practically, ensuring high reliability.

Keywords: classification, selection, mobile robots, rough sets, fuzzy sets

Streszczenie

Problem selekcji najlepszych obiektów może być rozwiązany z pomocą narzędzi wspomagających klasyfikację. Istnieje wiele narzędzi programowych przeznaczonych do eksploracji danych, w tym do klasyfikacji, ale każde z nich posiada pewne ograniczenia. Stosowane w nich metody selekcji nie sprawdzają się dla danych obciążonych błędami lub szumem, dla których istotna informacja powinna być dynamicznie pozyskiwana w trakcie trwania eksperymentu. Problem jest analizowany na przykładzie selekcji kandydatów na operatorów mobilnego robota (mobota). Praca zawiera porównanie podstawowych metod klasyfikacji (drzew klasyfikacyjnych, zbiorów przybliżonych oraz zbiorów rozmytych). Ponadto przedstawione jest nowe narzędzie programowe, skuteczne w selekcji kandydatów do wykonania trudnych zadań lub zawodów wymagających specyficznych predyspozycji. Prezentowany klasyfikator jest użyteczny zarówno w wypadku danych obciążonych szumami, jak i danych napływających dynamicznie, po rozpoczęciu procesu selekcji. Może być on wykorzystany w praktyce zapewniając dużą wiarygodność wyników.

Słowa kluczowe: klasyfikacja, selekcja, roboty mobilne, zbiory przybliżone, zbiory rozmyte

* Mgr inż. Barbara Łukawska, Katedra Informatyki, Wydział Automatyki, Elektroniki i Informatyki, Politechnika Świętokrzyska.

Paper submitted for publication: 10.01.2010

1. Introduction

Nowadays, processing a large set of data is nothing unusual but still it may be very complex. If the data is burdened with errors, noise or non-relevant information, the problem is even more complicated because extra questions arise such as what is the noise, which attributes should be analyzed and what kinds of dependencies are the most wanted. Data gathered for a choice of best candidates for a particular position is a good example here. Usually, the candidates are selected on the basis of data coming from personal interviews and tests. The first item of the data should disclose their predispositions but the second one their skills. Both types of the data may be imprecise. The candidates are classified, on the basis of the data, into two categories: good – *accepted* and bad – *rejected*. Hence, from algorithmic point of view the procedure of selection may be considered as a classification but likely using imprecise data.

A number of solutions for working with such kind of data have already been developed. Statistical methods are widely used in almost every area of science. There are many software tools applying these methods, such as calculation sheets available in most of the office software [1] (Microsoft Office Excel, OpenOffice.org Calc), where the data can be processed with the help of basic statistical functions, used to generate diagrams, etc. There are also many libraries for application developers and different programming languages with statistical functions [2]. Other types of software are R-type environments [3], SPSS software [4], Statistica [5] and Weka [6]. The latter offers many advanced data exploration techniques. Moreover, database systems, in addition to the data storage, offers data processing options.

Among classification methods, decision trees are the most frequently used [7, 8]. Decision trees are used in areas like medicine (diagnosis), computer sciences (data structures), botanic (classification) and psychology (decision theory).

Because data coming from humans may be incomplete or ambiguous, classical logical interpretation attempt may not be successful. In such case, rough sets [9, 10] and fuzzy sets [11, 12] are helpful.

Rough sets, along with decision tables, may be used to create decision rules useful in classification [13–15]. The advantage of rough sets is ability to find relations between object attributes, ability to data reduction and induction from upper and lower approximation of the decision rules. Approximation is a logic expression describing some and possible relationships between objects, their conditional and decision attributes. Rough sets are widely used in economics, medicine and computer sciences. There is also available software for this purpose, for example ROSETTA [16], RSES [17].

Many different classification methods use fuzzy sets [11, 18]. Traditional classification algorithms are enriched and become useful in analysis of real, usually not perfect objects (with missing data, ambiguity and/or conflicts) and linguistic variables. Fuzzy logic is widely used in modern sciences, such as technical and medical diagnosis. Because fuzzy logic offers good performance in real data processing, it is used in different expert and decision systems. Developing an application with fuzzy logic is easy with the help of libraries for different programming languages, like Free Fuzzy Logic Library for C++ [19], jFuzzyLogic for Java [20] or FuzzyCLIPS for CLIPS [21]. For working with fuzzy sets, mathematical tools like Simulink [22] may be helpful. There are also fuzzy logic based systems like fuzzyTECH [23].

Each of the methods listed above solves the problem of knowledge exploration in different way and offers different capabilities and results. However, choosing the most suitable method and initially preparing the data may be a problem. There is possible to test and compare some data mining techniques but usually only if they concern a particular data set. Actually, the most desirable is not a mathematical apparatus for data description, but a tool which may be used for real-life data classification. Unfortunately, none of the methods suits the task of mobot operator classification. The primary problem here is the need for dynamic data processing what means that the classification rules may change during a process of selection. The changes should be done continuously on the basis of incoming data because each new item of data may enrich our knowledge about a sense of "to be a good operator".

In the paper the problem is extensively investigated for selection of candidates for operators of mobile robot (mobot) guarding a given area [24–27]. Publications concerning the problem of training and selecting candidates are usually focused on military aspects [28]: tactical mobots, unmanned planes and helicopters, training of the recruits, pilots, mariners, etc. Medical uses of classification are also often found in the literature [29, 30] – for classification of diseases on the basis of symptoms. It is similar to the problem analyzed here. In publications concerning medical problems, rough sets theory and fuzzy sets theory are usually applied.

The paper is organized as follows. Section 2 contains a short description of our previous work. In section 3 the motivation is given. Section 4 describes the experiment with mobot operators. A novel approach to classification is given in section 5. The software classifier is presented in section 6. Section 7 contains conclusions from the research.

2. Previous work

The problem of selection may be solved with the help of tools for classification of objects. Among of them Weka [6, 31–33] is freely available software which offers different classification methods. Each of the methods uses different approach for the knowledge analysis, has different capabilities and gives different results.

At the very beginning of the research Weka was used for selection of the mobot operator candidates. Classification trees, rough sets and fuzzy sets were chosen for this purpose. Candidates were classified with "the train-and-test" method. Data from the experiment with mobot operator training (see chapter 4) was a basis for the classification. To make the comparison possible, Q function was defined. Q reflects the monitoring quality of a particular operator. Candidates with the highest Q should be chosen for the *accepted* group.

Weka creates a model on a basis of training data (the set of objects supplemented with information about destination group). The model (classification tree or the set of decision rules) may be then used for classification of new objects.

The classification was repeated with different training sets (chosen arbitrary) and different classification methods with different parameters. Results were compared so as to find the most appropriate method. The conclusions are summarized in Table 1.

Table 1

Comparison of the classification methods

	Classification trees	Rough sets	Fuzzy sets
Classification basis	Expert knowledge collected in a training set	Expert knowledge collected in a decision table	Grouping
The kind of computed dependencies	Not-equal conditions for each variable	Equal conditions for each variable	Grouping and 'then' rules based on the group membership
Results correctness	Acceptable	Acceptable	Invalid results – sometimes all candidates are rejected
Classification parameters influence	Big – depending on the algorithm chosen, the correctness from 78.2051% to 88.4615%, depending on the training set choice, from 73.5537% to 96.5517%	Big – depending on the digitization level set, the number of reducts and rules changes, depending on the training set choice, correctness from 73.5537% to 100%	
Generation time	Negligible	May be significant	Negligible
Static selection process	Yes – increasing the training set may totally change the resulting tree	Partially – increasing the training set creates new rules, and may change the reduct set	Yes – increasing the training set may totally change grouping rules
Main advantages	High memory and computation efficiency, no limitations on the object complexity, able to analyze many number of variables, possibility to analyze mixed type variables	Possible to restrict the number of attributes with the reduct set, possibility to analyze large number of variables, possibility to analyze mixed type variables	Due to totally invalid results, further analysis was canceled
Main disadvantages	Static only, requires collecting and processing all data for the training set, possibility to test single attributes without dependencies, problem with additional rules for the number of group members (object classes), problem with object arrangement in a given category	Static only, requires collecting and processing all data for the decision table, possibility to test single attributes without dependencies, problem with additional rules for the number of group members (object classes), problem with object arrangement in a given category, problem with unclassified elements	

The calculations based on decision trees and rough sets were quite acceptabl. If correct parameters were chosen, classification correctness reached nearly 100%. In the worst case it was 73.5%, what meant that about 1/4 of the candidates were classified incorrectly.

The methods use basic algorithms from their domains. Not all capabilities of a particular method were exploited (e.g. there are "fuzzy decisions trees", where a classical decision tree is combined with fuzzy logic). Nevertheless, the software is freely available and easy to use, what is its big advantage.

3. Motivation

Most of the software requires already collected, two complete sets of data. Rules for a given training sample (supplemented with expert knowledge) are created, and then used for a test sample. If the training sample is not adequate for the test sample or the training sample is wrongly chosen, the resulting rules are inaccurate. Furthermore, a diversity of the objects, as well our knowledge about the objects may change during the classification. The problem is even more complicated if the test sample should not be divided into groups, but a fixed size group of the best candidates must be found. An example is to find 10 patients with the highest temperature, ignoring the fact that a patient with the temperature over 38 degrees has a fever.

Summarizing, none of the methods suits the task of mobot operator classification. The primary problem is the need for dynamic data processing, where the classification rules may change during the process. The changes are made on the basis of incoming data. Hence, the selection procedure should be dynamic, accepting continuously incoming data. Particularly, the system should have a practical use for selecting the best candidates for a given position, choosing the best objects, etc. It should work dynamically and allow for changing classification rules on-the-fly, so that to speed up the process of selection and make it simpler. In case of a multi-stage selection procedure where choices are after each of the steps, it should make the selection cheaper while making the process only slightly more complicated.

4. Simulation experiment

For training and ranking operators of different complex machines, virtual reality simulators are frequently used. Such solution is cheaper and safer than working on a real machine. It makes it possible to give proper results (a group of highly qualified operators) and keep the costs adequately low [34, 35].

Virtual reality simulator is used for training and ranking mobot operators [24, 26, 27, 36]. The simulator is similar to a flight simulator. Virtual mobot controlled by an operator[1] moves in a closed room (so called "scene"). The operator tries to find changes in the environment using static map of initial scene and photos taken by the mobot on his demand. The simulator test consists of four stages each one comprises a set of scenes. Each stage has different purpose and different attributes are tested: overall predispositions, learning ability, strategic ability and stress resistance.

The simulator collects information about activities of the candidate. Next, these activities are analyzed so that his predispositions to work as the operator are measured.

[1] Students of Kielce University of Technology played a role of candidates for mobot operators.

To summarize and compare the results, function called "monitoring quality" (Q) is defined. Q is computed from: the number of changes detected in the environment, the number of errors, the number of moves and the number of pictures taken. On this basis there is possible to find the best candidate (or a subgroup) from the whole group.

A lot of interesting data, concerning controlling a mobot, was collected in our experiment. On the basis of this data, the following conclusions were drawn [25, 27, 36–38]:

– the monitoring quality is an individual attribute of an operator;
– the monitoring quality may be improved during training in the virtual reality;
– the simulator allows for training candidates for mobot operators;
– analyzing Qs allows for ranking the candidates;
– thanks to the multi-stage training connected with selection, it is possibly to significantly reduce costs of the training.

However, training a large group of candidates and choosing the best is time-consuming, expensive and hard to execute even with the help of the simulator. Obviously, the possibility of elimination of persons with lowest skills during the training would make it cheaper. It was proven, that rejecting weakest candidates after each stage gives results similar to choosing the best candidates after the whole training. That is why such selection procedure is reliable and gives quite good results [25]. A dynamic classifier supporting the procedure should highly improve the selection and reduce the costs.

5. Dynamic classification

Most of studied classification methods (and software tools) approach the problem of data classification in a static way. In case of a problem like mobot operator classification there is a need for some dynamic selection procedure. It should be connected with virtual simulator training. Data from the experiment should be analyzed on-the-fly, and its results should alter the course of the experiment. This could be achieved by repeated (after each stage of the training) dynamic selection of candidates for the best group for further training and rejecting the rest of the candidates. So, a classifier connected with the virtual mobot simulator should select a group of most skilled candidates. Because candidates may start the training in different moments, the analysis should be started for incomplete set of data.

Because the data come from simulation experiment with humans, there is a high possibility that the data will be incomplete or ambiguous. Thus the selection procedure should use fuzzy set theory.

6. Classifier

6.1. Overview

To be flexible, the classifier works on objects representing the candidates. It is able to rank and classify objects and create their models. Full information about every object is stored[2].

[2] The classifier is a C++ application, developed with the help of wxWidgets framework.

An object representing a candidate is described by grouped sets of elements. A group corresponds to one stage of the training. Each element consists of attributes. Each of the attributes is supplemented with bell-type membership function[3], in which coefficients (*a*, *b* and *c*) are user-defined. An object is ranked according to the attribute values and coefficients. They determine a so called *grade* of the object. The grade is computed automatically as soon as required data is collected. It is a value of membership function, computed on the basis of parameter values for a given attribute.

The following information is gathered:

- For an element:
  - a value of additional, predefined attribute *Q*, which is a sum of all values of the attributes multiplied by their weight.
- For a group:
  - average values of all attributes, these may be used to compute the grade;
  - average value of *Q* attribute that may be used to compute the grade;
  - *Q* attribute gap that may be used to compute the grade;
  - group grade what is an algebraic product of grades of chosen (or all) attributes, and may include gap and *Q*; this summary information may be used in selection;
- Global:
  - average values of all attributes, these may be used to compute the grade;
  - a gap computed as a weight average from group gaps;
  - the grade computed as an arithmetic average of the chosen (or all) groups.

6.2. Selection

The selection may take different values into account. It relies on choosing specific (direct or percentage) number of objects achieving maximum values for a given attribute. An object may be considered, if gains positive grade on other attributes, given by the user.

The classifier allows for summarizing data on different planes. Attribute values of the worst from chosen candidates or the best from *rejected* candidates may be used as a decision rule for classifying new candidates. These values are updated dynamically, as a new item of data arrives. That is why the model is more flexible and precise.

Candidate ranking should be primarily used for selection. From the other hand, there are situations where the grade is used to modify the training, not for rejecting the weakest candidates. An example may be the problem of training all the candidates to a given level, without selecting the best. For such a case, lower rank means more time spent on training to reach required skill level.

On the basis of the final rank, the following decisions may be taken:

- In case of the selection, the grade may be used to eliminate the worst candidates after each step of the training. In such case, the cost of training may be significantly reduced (by decreasing the number of the candidates), still keeping dependability high.
- In case of training all of the candidates to a given level, weak persons are forced to spend more time on training. Persons not achieving the required results may be moved backward to repeat some stage(s).

[3] $\mu_A(x; a, b, c) = 1/(1+(|(x-c)/a|)^{2b})$

6.3. Models

On the basis of currently available data, the best, the worst and average objects are created on-the-fly. The best ('MDL-BDB'), the worst ('MDL-ZLY') and average ('MDL-SR') objects are identical as ordinary objects, but their values are updated automatically as new data is entered.

Values of each attribute are compared with appropriate membership function coefficients. On this basis the best values (closest to the *c* coefficient of the membership function) and the worst values (farthest from the *c* coefficient) are chosen. Values of the 'MDL-SR' object are arithmetic averages of appropriate attributes of all valid elements.

Additionally, a history of model stabilization is created, on the basis of global object grade. To accomplish this, minimal, maximal and average grade for complete objects are computed, just as any object is completed. These values create the model. Consecutive models are stored in a list, and are presented in a form of table and diagram. A model is acclaimed as stable, if in a set of given number of consecutive models, the difference of grade for each model pair is not greater than a given value.

So, the classifier has two working modes: predictive model is created during the classification, and descriptive model is created on-the-fly. These two models supplement each other. Theoretically the best object may be used to create classification rules for other objects.

6.4. Auto tuning

The model of theoretically the best object 'MDL-BDB' may be used to scale the grades.

In real-life applications, theoretically the best value may never be achieved. In such case, none of the objects may achieve good enough grades. Sometimes such situation is correct, but sometimes there is a need to scale grades to reflect real constraints. That is why there is an option to tune automatically 'c' coefficient of the membership function. In such case, as 'MDL-BDB' model changes, 'c' coefficient for each attribute is set to average value from the 'MDL-BDB' object, what results in updating all the grades.

6.5. Ranking lists

Raking lists are created from completed objects. These lists may contain objects sorted according to:

- global average *Q*;
- global grade;
- the distance from the 'MDL-BDB' object, that is computed as a distance in multi-dimensional space between a given object and the 'MDL-BDB' object.

6.6. Advantages of the classifier

The classifier has two modes of operation:

- *User classification* requires that the user has some expert knowledge. Using this knowledge, the user chooses ranking parameters and boundaries directly. Thus, that the predictive model is user-defined;
- *Automatic classification* where the parameters are computed automatically on the basis of the model being created. The user must predict what values particular attributes may reach.

In the first mode, if correct parameters were chosen, selection of the given number of the best candidates is possible. Unfortunately, choosing bad values may lead to empty group.

In the second mode descriptive model is used to compute predictive model, thus the classification rules.

The selection procedure has a form of progressive selection after each stage. Primary advantages of such an approach are the following:

- it is a dynamic procedure, and can alter the experiment;
- it may be applied in any moment, so no complete data set before the selection is required. The selection may be performed if some of the candidates finished given stage. A candidate being initially in "the best" group may be rejected if better candidate is found;
- the process of creating and stabilizing models may become a basis to draw hypotheses about another classification process for different series of data;
- auto-tuning makes it possible to choose better membership function and more precise ranking of the candidates;
- it is possible to test different versions, variants, and influence of each parameter on the results. Optimal scheme for training and selection may be chosen then.

Because the membership function is used to compute grades, it is possible to scale non-linear values into $\langle 0; 1 \rangle$ range. Moreover, it is possible to easily extend the range, where the membership function reaches 0 or 1. It is very important, if attribute values are set intuitively, with possible errors. Due to the fact that the basic membership function is similar to normal distribution, it fits goodly experimental data.

This is very important for analysis of data burdened with errors and measurement inaccuracies. Automatic scaling allows for pinpointing the hypothetical range, leaving some space for fluctuations (just as for extending the range). Using the membership function makes it possible to control the headcount easily. It is possible because apart from "yes-no" information generated by basic classification methods (classification trees, rough sets), there is also "maybe" information with the "goodness" level. Fuzzy, soft transition leads to possible exchange of good objects at the border. The model is more flexible, and makes it possible to easy move objects between categories, without the need of altering the model.

The model is created dynamically. Successive objects update its attributes. Analysis of the stabilization history, especially on the stability aspect, allows for checking the model adaptation.

Additional functions make it possible to summarize the data in different way. The comparison of real results with the theoretical results allows for drawing conclusions about the classification assumptions that may be used to modify these assumptions. A comparison of models in different phases allows for evaluating the learning scheme, thus the training itself. Analysis of the stabilization process allows for drawing some conclusions about the optimal number of candidates. Comparison of the ranking lists may be used to evaluate correctness and usefulness of the Q function, and the method of raking, too.

7. Conclusions

Most of the data mining software deals with data classification in a static way. In finding the best operator candidates there is a need for some dynamic data exploration during the course of the training.

A classifier fulfilling presented assumptions implements a dynamic selection procedure. The classifier is able to rank, classify objects and compute its models. Basic assumptions for the classifier are the following:

- it works with "objects" (a candidate is an object) for improving flexibility;
- it works dynamically, new objects may arrive while the analysis takes place;
- instead of a single attribute, an object may be described with a set of attributes;
- an object may be ranked on the basis of chosen attributes;
- the total grade is computed on the basis of a few attributes and further processed;
- group of the best candidates may be selected quantitative and not qualitative only (e.g. 10 of the best objects – not objects with some attribute above given threshold value);
- on the basis of data, models for the best, the worst and average object are created on-the-fly;
- the history of model stabilization is created.

Such an attempt leads to lower costs and makes the selection easier to organize (there is no need to collect all the data before selection). Simultaneously it gives good results for current sample, and allows for foreseeing some rules. The classifier does not create a mathematical apparatus itself, but it makes it possible to have some practical use.

Dynamic, multi-stage selection procedure may be created with the help of the classifier. The test may consist of a few stages, where the worst candidates are dropped after each stage. Such selection procedure could be applied for training operators of different complex machines, devices, etc.

References

[1] Dobre Programy (dobreprogramy.pl).
[2] ICM Centrum Obliczeniowe (www.icm.edu.pl/kdm/Numeryka:_Statystyka).
[3] The R Project for Statistical Computing (www.r-project.org).
[4] SPSS (www.spss.com/spss).
[5] StatSoft Polska – Statistica (www.statsoft.pl).
[6] The University of Waikato – WEKA (www.cs.waikato.ac.nz/ml/weka).
[7] Cichosz P., *Systemy uczące się*, WNT, 2000.
[8] Koronacki J., Ćwik J., *Statystyczne systemy uczące się*, WNT, 2005.
[9] International Rough Set Society (roughsets.home.pl/www).
[10] Rough Set Database System (rsds.univ.rzeszow.pl).
[11] Rutkowski L., *Metody i techniki sztucznej inteligencji*, Wydawnictwo Naukowe PWN, 2006.
[12] Zadeh L.A., *Home page* (www.cs.berkeley.edu/~zadeh).
[13] Pawlak Z., *Rough classification*, International Journal of Human-Computer Studies, Volume 51, Issue 2, 1999.
[14] Polkowski L., Tsumoto S., Lin T.Y., *Rough Set Methods and Applications*, Physica-Verlag, Heidelberg, Germany 2000.

[15] Tsumoto S., *Modelling medical diagnostic rules based on rough sets*, RSCTC, LNCS (LNAI), Vol. 1424, Springer, Heidelberg, 1998, 475-482.
[16] ROSETTA: A Rough Set Toolkit for Analysis of Data (rosetta.lcb.uu.se/general/features.htm).
[17] Rough Set Exploration System (logic.mimuw.edu.pl/~rses).
[18] Kuncheva L.I., *Fuzzy Classifier Design*, Springer-Verlag, Heidelberg 2000.
[19] Free Fuzzy Logic Library (ffll.sourceforge.net).
[20] FuzzyLogic: Open Source Fuzzy Logic (jfuzzylogic.sourceforge.net).
[21] Fuzzy Logic in Integrated Reasoning (www.iit.nrc.ca/IR_public/fuzzy).
[22] Simulink – Simulation and Model-Based Design (www.mathworks.com/products/simulink).
[23] FuzzyTECH Home Page (www.fuzzytech.com).
[24] Sapiecha K., Bedla M., Łukawska B., Paduch P., *Computer-based system for training and selecting mobile robot operators – evolving software tools*, Informatyka – badania i zastosowania (IBIZA), Kazimierz Dolny 2007.
[25] Sapiecha K., Łukawska B., Bedla M., *Computer-based system for training and selecting mobile robot operators – selection procedure*, Informatyka – badania i zastosowania (IBIZA), Kazimierz Dolny 2007.
[26] Bedla M., Łukawska B., Sapiecha K., *Software architecture for a system of remote training mobot operators*, Proceedings of the 3-rd International Conference: ACSN 2007, Lwów 2007.
[27] Łukawska B., Paduch P., Sapiecha K., *An application of virtual reality for training and ranking operators of mobile robot*, Annales UMCS – Informatica AI 5, Lublin 2006, 393-399.
[28] International Military Testing Assocation (www.internationalmta.org/2003/index.htm).
[29] Duch W., *Sztuczna Inteligencja w Medycynie* (www.fizyka.umk.pl/~duch/ref/PL/_9ai-med/ai-med2.html#przyklad).
[30] FuzzyFluid (www.nickels.fi/likvor/likvor.htm).
[31] Weka Docs, An online help for free data-mining software (wekadocs.com).
[32] Weka API Help (weka.sourceforge.net/doc).
[33] Weka-related Projects (www.cs.waikato.ac.nz/~ml/weka/index_related.html).
[34] Balachandran A., Rabuya L.C., Shinde S., Takalkar A., *Introduction to modeling and simulation systems: Historical perspectives* (www.uh.edu/~lcr3600/simulation/historical.html).
[35] Alexander A.L., Bruny'e T., Sidman J., Weil S. A., *From gaming to training: A review of studies on fidelity, immersion, presence, and buy-in and their effects on transfer in pc-based simulations and games*, 2005.
[36] Sapiecha K., Łukawska B., Paduch P., *Experimental Data Driven Robot for Pattern Classification*, Annales UMCS Informatica AI 3, Lublin 2005, Vol. III, 263-271.
[37] Sapiecha K., Łukawska B., Bedla M., *A procedure of verification of comprehensive tests for selection of candidates for operators of mobile robot*, Annales UMCS Informatica AI VIII, 2 Lublin 2008, 97-106.
[38] Łukawska B., Łukawski G., Sapiecha K., *Experimental evaluation of two touring simulators for training operators of mobot*, Czasopismo Techniczne, z. 1-I/2008, Wydawnictwo PK, Kraków 2008.