KEEL: A data mining software tool integrating genetic fuzzy systems

Jesús Alcalá-Fdez, Salvador García, Francisco José Berlanga
Alberto Fernández, Luciano Sánchez, M.J. del Jesus and Francisco Herrera

Abstract— This work introduces the software tool *KEEL* to assess evolutionary algorithms for Data Mining problems including regression, classification, clustering, pattern mining and so on. It includes a big collection of genetic fuzzy system algorithms based on different approaches: Pittsburgh, Michigan, IRL and GCCL. It allows us to perform a complete analysis of any genetic fuzzy system in comparison to existing ones, including a statistical test module for comparison. The use of KEEL is illustrated through the analysis of one case study.

I. INTRODUCTION

One of the most popular approaches in the Computational Intelligence community is the hybridization between fuzzy logic [1] and genetic algorithms (GAs) [2], [3] leading to genetic fuzzy systems (GFSs) [4]. A GFS is basically a fuzzy system augmented by a learning process based on evolutionary computation, which includes GAs, genetic programming and evolutionary strategies among other evolutionary algorithms (EAs) [5].

Fuzzy systems are one of the most important areas for the application of the Fuzzy Set Theory. Usually it is considered a model structure in the form of fuzzy rule based systems (FRBSs). FRBSs constitute an extension to classical rule-based systems, because they deal with "IF-THEN rules, whose antecedents and consequents are composed of fuzzy logic statements, instead of classical ones.

The automatic definition of an FRBS can be seen as an optimization or search problem, and GAs are a well known and widely used global search technique with the ability to explore a large search space for suitable solutions only requiring a performance measure. Moreover, the generic code structure and independent performance features of GAs make them suitable candidates to incorporate a priori knowledge. These capabilities extended the use of GAs in the development of a wide range of approaches for designing FRBSs over the last few years.

The use of GFSs in problem solving is a widespread practice. Problems such as engineering applications [6], ecology [7], medicine [8] or robotic [9] show their suitability as problem solvers in a wide range of scientific fields.

Although GFSs are powerful for solving a wide range of scientific problems, their use requires a certain programming

J. Alcalá-Fdez, S. García, A. Fernández and F. Herrera are with Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain {jalcala,salvagl,alberto,herrera}@decsai.ugr.es
F.J. Berlanga and M.J. del Jesus are with the Department of Computer Science, University of Jaén, 23071 Jaén, Spain {berlanga,mjjesus}@ujaen.es
L. Sánchez is with the Department of Computer Science, University of Oviedo, 33204 Gijón, Spain luciano@uniovi.es

expertise along with considerable time and effort to write a computer program for implementing the often sophisticated algorithm according to user needs. This work can be tedious and needs to be done before users can start focusing their attention on the issues that they should be really working on. In the last few years, many fuzzy logic software tools have been developed to reduce this task. Although a lot of them are commercially distributed (for example, MATLAB Fuzzy logic toolbox[1]), a few are available as open source software (we recommend visiting the Fuzzy Sets and Systems Software Repository[2] for an overall view of most of them). Open source tools can play an important role as is pointed out in [10].

In this contribution we introduce a non-commercial Java software tool named *KEEL* (Knowledge Extraction based on Evolutionary Learning)[3]. This tool empowers the user to assess the behaviour of EAs for different kinds of Data Mining (DM) problems: regression, classification, clustering, pattern mining, etc. Consequently, the application of EAs for learning fuzzy systems is also included in KEEL, including a representative set of GFSs. It allows us to perform a complete analysis of any genetic fuzzy system in comparison to existing ones, including a statistical test module for comparison.

This tool can offer several advantages:

- First of all, it reduces programming work. It includes a big library with GFS algorithms based on different paradigms (Pittsburgh, Michigan, IRL and GCCL) and simplifies their integration with different pre-processing techniques. It can alleviate researchers from the mere "technical work" of programming and enable them to focus more on the analysis of their new learning models in comparison with the existing ones.
- Secondly, it extends the range of possible users applying GFSs. An extensive library of algorithms together with easy-to-use software considerably reduce the level of knowledge and experience required by researchers in evolutionary computation and fuzzy logic. As a result researchers with less knowledge, when using this framework, would be able to apply successfully these algorithms to their problems.
- Third, due to the use of a strict object-oriented approach for the library and software tool, these can be used on any machine with Java. As a result, any researcher

[1]http://www.mathworks.com
[2]http://www.fuzzysoftware.org
[3]http://www.keel.es

can use KEEL on his machine, independently of the operating system.

In order to describe KEEL, this work is arranged as follows. The next section introduces the different genetic learning coding approaches according to the way of coding a rule base (RB). Section III presents the GFSs in KEEL and points out the future algorithms in KEEL. Section IV describes KEEL and its main features and modules. In Section V, one case study is given to illustrate how KEEL should be used. Finally, Section VI points out some conclusions and future work.

II. GENETIC LEARNING

Although GAs were not specifically designed for learning, but rather as global search algorithms, they offer a set of advantages for machine learning. Many methodologies for machine learning are based on the search of a good model inside the space of possible models. In this sense, they are very flexible because the same GA can be used with different representations. Genetic learning processes cover different levels of complexity according to the structural changes produced by the algorithm, from the simplest case of parameter optimization to the highest level of complexity for learning the rule set of a rule-based system, via the coding approach and the cooperation or competition between chromosomes.

When considering the task of learning rules in a rule based system, a wider range of possibilities is open. When considering a rule based system and focusing on learning rules, the different genetic learning methods follow two approaches in order to encode rules within a population of individuals:

- The "Chromosome = Set of rules", also called the Pittsburgh approach, in which each individual represents a rule set [11]. In this case, a chromosome evolves a complete RB and they compete among them along the evolutionary process. GABIL is a proposal that follows this approach [12].
- The "Chromosome = Rule" approach, in which each individual codifies a single rule, and the whole rule set is provided by combining several individuals in a population (rule cooperation) or via different evolutionary runs (rule competition).

  In turn, within the "Chromosome = Rule" approach, there are three generic proposals:

  - The Michigan approach, in which each individual encodes a single rule. These kinds of systems are usually called learning classifier systems [13]. They are rule-based, message-passing systems that employ reinforcement learning and a GA to learn rules that guide their performance in a given environment. The GA is used for detecting new rules that replace the bad ones via the competition between the chromosomes in the evolutionary process. An interesting study on the topic can be found in [14].
  - The IRL (Iterative Rule Learning) approach, in which each chromosome represents a rule. Chro-

mosomes compete in every GA run, choosing the best rule per run. The global solution is formed by the best rules obtained when the algorithm is run multiple times. SIA [15] is a proposal that follows this approach.
  - The GCCL (genetic cooperative-competitive learning) approach, in which the complete population or a subset of it encodes the RB. In this model the chromosomes compete and cooperate simultaneously. COGIN [16] and LOGENPRO [17] are examples with this kind of representation.

These four genetic learning approaches (Pittsburgh, Michigan, IRL and GCCL) have been considered for learning RB, and we can find different examples of them with fuzzy or interval rules in KEEL[4]. For example, in the case of the interval rules based systems for classification, we can find the following algorithms in KEEL:

- Pittsburgh approach: GAssist [18].
- Michigan approach: XCS [19], UCS [20].
- IRL approach: SIA [15], Hider [21].
- GCCL approach: LOGENPRO [17].

III. GENETIC FUZZY ALGORITHMS IN KEEL

In the last few years we observe the increase of published papers in the topic due to the high potential of GFSs. Contrary to neural networks, clustering, rule induction and many other machine learning approaches, GAs provide a mechanism to encode and evolve rule antecedent aggregation operators, different rule semantics, RB aggregation operators and defuzzification methods. Therefore, GAs remain today as one of the fewest knowledge acquisition schemes available to design and, in some sense, optimize FRBSs with respect to the design decisions, allowing decision makers to decide what components are fixed and which ones evolve according to the performance measures.

KEEL divide the GFS approaches into two processes, tuning and learning:

- Genetic tuning. If there exists a knowledge base (KB), we apply a genetic tuning process for improving the FRBS performance.
- Genetic learning. The second possibility is to learn KB components (where we can even include an adaptive inference engine).

We can classify the GFSs in KEEL according to these two processes and according to the FRBS components involved in the genetic learning process [22]. We can find different examples of them for classification and regression in KEEL[4]. For example, the following algorithms are available:

- Genetic tuning of KB parameters: a tuning method for obtaining high-performance fuzzy control rules by means of special GAs [23], etc.
- Genetic Rule Weight Learning: a GA for learning rule weight and selecting rules [24], etc.

[4]http://www.keel.es/algorithms.php

- Genetic RB learning: The pioneer Pittsburgh based method proposed by Thrift [25], SLAVE [26], COR-GA [27], a hybrid algorithm of Michigan and Pittsburgh fuzzy genetics-based machine learning approaches [28], etc.
- Genetic TSK FRBS learning: a two-stage genetic FRBS [29], an evolutionary process with a local identification of prototypes to obtain the set of initial local semantics-based TSK rules [30], etc.
- Simultaneous genetic learning of KB components: a multi-stage hybrid GA-evolution strategy process for designing approximate FRBS [31], etc.
- Genetic rule selection: the first contribution in this area [32], etc.

At the moment, we are developing a new set of GFSs to include in KEEL.

IV. KEEL DESCRIPTION

KEEL is a software tool to assess EAs for DM problems including regression, classification, clustering, pattern mining and so on. The presently available version of KEEL consists of the following function blocks (see Fig. 1):
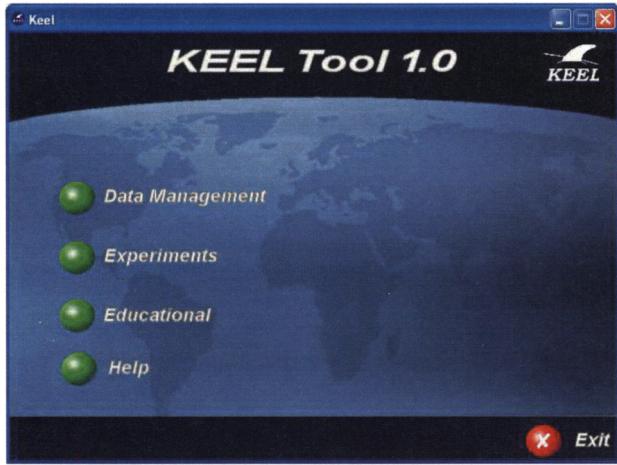

Fig. 1.   Screenshot of the main window of KEEL software tool

- *Data Management*: This part is made up of a set of tools that can be used to build new data, to export and import data in other formats to or KEEL format, data edition and visualization, to apply transformations and partitioning to data, etc.
- *Design of Experiments*: The aim of this part is the design of the desired experimentation over the selected data sets and providing for many options in different areas: type of validation, type of learning (classification, regression, unsupervised learning), etc.
- *Educational Experiments*: With a similar structure to the previous part, this allows for the design of experiments that can be run step-by-step in order to display the learning process of a certain model by using the software tool for educational purposes.

This structure makes KEEL software useful for different types of user, who expect to find different functionalities in

a DM software. Shortly, we describe the main features of KEEL:

- EAs are presented in predicting models, pre-processing (evolutionary feature and instance selection) and post-processing (evolutionary tuning of fuzzy rules).
- It includes data pre-processing algorithms proposed in specialized literature: data transformation, discretization, instance selection and feature selection.
- It has a statistical library to analyze algorithms' results. It comprises a set of statistical tests for analyzing the suitability of the results and performing parametric and non-parametric comparisons among the algorithms.
- Some algorithms have been developed by using Java Class Library for Evolutionary Computation (JCLEC) [33].
- It provides a user-friendly interface, oriented to the analysis of algorithms.
- The software is aimed to create experimentations containing multiple data sets and algorithms connected among themselves to obtain an expected results. Experiments are independently script-generated from the user interface for an off-line run in the same or other machines.
- KEEL also allows creating experiments in on-line mode, aiming an educational support in order to learn the operation of the algorithm included.

In the following, we will briefly describe the main features of the *Design of Experiments* function block. It is a Graphical User Interface that allows the design of experiments for solving different machine learning problems. Once the experiment is designed, it generates the directory structure and files required for running them in any local machine with Java (see Fig. 2).
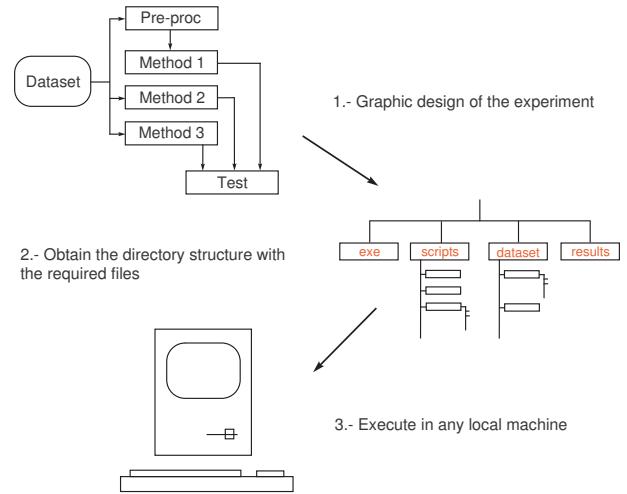

Fig. 2.   Design of experiments

The experiments are graphically modeled, based on data flow and represented by graphs with node-edge connections. It allows us to choose the type of validation (k-fold cross validation or 5x2 cross validation) and type of learning (regression, classification or unsupervised).

Then, we have to select the data sources, drag the selected methods into the workspace and establish connections between methods and data sets. Also, the addition of statistical analysis of results is supported by including the corresponding techniques. In any moment, each component added in the experiment can be configured by double-clicking in the respective node. Fig. 3 shows an example of an experiment following the MOGUL methodology [34] in classification and using a report box to obtain a summary of results. The configuration window of the MOGUL method is also shown in this figure.
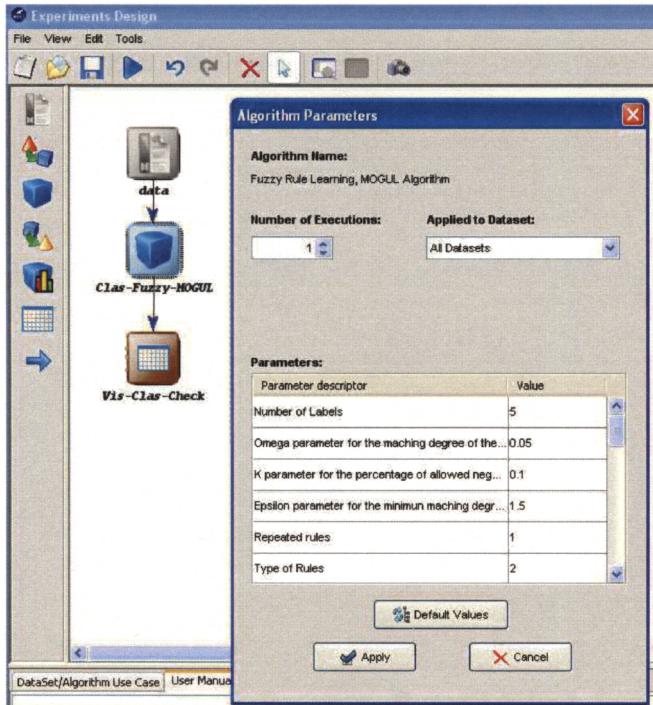
Fig. 3. Example of an experiment and the configuration window of a method.

When the experiment has been designed, the user can choose either to save the design in XML file or obtain a zip file. The latter will contain the directory structure and required files for running the experiment in an independent machine with Java. The zip file will include the data sources, jar files of the algorithms, configuration files in XML format and a Java Tool, named *RunKeel*, to run the experiment. *RunKeel* can be seen as a simple scripting environment that interprets the script file in XML format, runs all the indicated algorithms and saves the results in one or several report files.

V. EXPERIMENTAL EXAMPLE

In this section, we present a case study as example of the functionality and process of creating an experiment in the KEEL software tool. This study is focused on the development of a comparison of two fuzzy rule based learning algorithms and a subsequent analysis of the results in *12 problems for classification*. These problems are summarized in Table I.

TABLE I
DATA SETS SUMMARY DESCRIPTIONS.

Data set	#Examples	#Atts.	#Classes
Bupa	345	6	2
Cleveland	297	13	5
Ecoli	336	7	8
Glass	214	9	7
Haberman	306	3	2
Iris	150	4	3
Monk-2	432	6	2
New-thyroid	215	5	3
Pima	768	8	2
Vehicle	846	18	4
Wine	178	13	3
Wisconsin	683	9	2

Methods considered for the experiment are SLAVE algorithm [26] and Chi et al. algorithm [35] with rule weights [36] (we denote it as Chi-RW algorithm). SLAVE algorithm is a RB inductive learning algorithm based on the IRL approach and Chi-RW algorithm is the adaptation of the Wang-Mendel's algorithm [37] to the classification problem using weights in the consequent of the rules.

To develop the experiment we consider a *10-folder cross-validation model*. For each one of the 10 data partitions, SLAVE algorithm has been run 5 times since this method is probabilistic (a total of 50 runs). Moreover, a pairwise Wilcoxon test was applied in order to ascertain if differences in the performance of the methods are significant.

The initial linguistic partitions are comprised by *three linguistic terms* with uniformly distributed triangular membership functions giving meaning to them. The values considered for the input parameters of each method are:

SLAVE parameters:
- Population Size = 100
- Number of Iterations Allowed without Change = 500
- Mutation Probability = 0.01
- Crossover Probability = 1.0 (it is always applied)

Chi-RW algorithm parameters:
- Computation of the compatibility degree: Product T-norm.
- Rule Weight: Penalized Certainty Factor.
- Combination of the compatibility degree and the rule weight: Product T-norm.
- Inference method: Classic method (winning rule).

To do this experiment in KEEL, first of all we click the Experiment option in the main menu of the KEEL software tool, define the experiment as a Classification problem and use a 10-fold cross validation procedure to analyze the results. Next, the first step of the experiment graph set-up is to choose the data sets of the Table I to be used. The partitions in KEEL are static, allowing that further experiments carried out will give up being dependent on particular data partitions. Optionally, an user could create his own partitions with the function block *Data Management*.

The graph in Fig. 4 represents the flow of data and results from the algorithms and statistical techniques. A node can represent an initial data flow (group of data sets), a
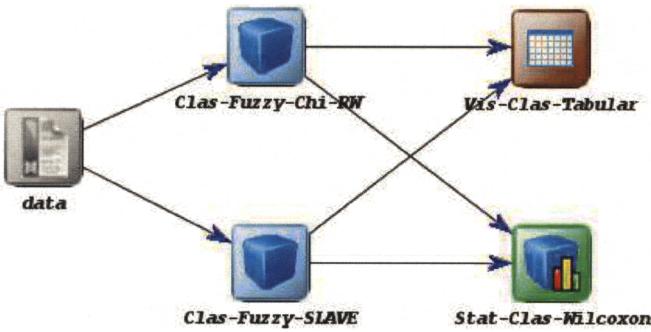
Fig. 4.   Experiment graph of the case study

Dataset	CHI-RW			SLAVE		
	Acc_{Tr}	Acc_{Tst}	#Rules	Acc_{Tr}	Acc_{Tst}	#Rules
Bupa	59.87	57.87	43.3	60.60	58.28	3.9
Cleveland	91.25	39.09	230.5	79.05	53.52	39.6
Ecoli	79.53	78.33	43.5	82.75	79.10	11.7
Glass	65.99	60.04	27.1	71.70	62.16	12.8
Haberman	74.26	73.19	16.7	74.90	74.35	2.9
Iris	93.78	94.00	14.7	96.98	94.86	3.3
Monk-2	100.0	48.84	301.8	67.36	67.23	1.3
New-thyroid	85.94	84.24	18.4	89.82	87.99	3.9
Pima	75.62	72.40	105.2	75.45	74.44	4.7
Vehicle	65.92	60.77	227.8	66.31	60.18	20.8
Wine	98.75	92.68	121.1	94.60	90.42	4.3
Wisconsin	98.08	91.21	224	97.16	95.72	5.1
Average	**82.42**	71.06	114.51	79.72	**74.85**	**9.53**

pre-process/post-process algorithm, a learning method, test or a visualization of results module. They can be easily distinguished according the color of the node. All their parameters can be adjusted by clicking twice on the node (see Section IV). Notice that KEEL incorporates the option of configure the number of runs for each probabilistic algorithm, including this option in the configuration dialog of each node (5 in the case of SLAVE algorithm). Logically, directed edges connecting two nodes represent a relation between them (data or results interchange). When the data is interchanged, the flow includes pairs of train-test data sets. Thus, the graph in this specific example describes a flow of data from the 12 data sets to the nodes of the two learning methods (*Clas-Fuzzy-Chi-RW* and *Clas-Fuzzy-SLAVE*).

After the models are trained, the instances of the data set are classified. These results are the inputs for the visualization and test modules. The module *Vis-Clas-Tabular* receives these results as inputs and generates output files with several performance metrics computed from them, such as confusion matrices for each method, accuracy and error percentages for each method, fold and class, and a final summary of results. Fig. 4 also shows another type of results flow, the node *Stat-Clas-Wilcoxon* which represents the statistical comparison, results are collected and a statistical analysis over multiple data sets is performed by following the indications given in [38]. The test used in this example is a pairwise Wilcoxon test.

Once the graph is defined, we can set up the associated experiment and save it as zip file for an off-line run. Following the structure of directories shown in Fig. 2, the experiment is set up as a set of XML scripts and a jar program for running it. Within the *results* directory, there will be directories used for housing the results of each method during the run. For example, the files allocated in the directory associated to a fuzzy learning algorithm will contain KB. In case of a visualization procedure, its directory will house the results files. The results obtained by the analyzed methods are shown in Table II, where #R stands for the averaged number of rules, Acc_{tra} and Acc_{tst} respectively for the averaged accuracy obtained over the training and test data.

In case of a test procedure, the *Stat-Clas-Wilcoxon* directory will house the results files. Test modules in KEEL provide the output in text and LaTeX formats. They generate

associated tables with information of the statistical procedure. In this case, Tables III and IV are generated by the *Stat-Clas-Wilcoxon* module when the run is finished.

		N	Mean Rank	Sum of Ranks
SLAVE vs. Chi-RW	Positive Ranks	10	6.9	69.0
	Negative Ranks	2	4.5	9.0
	Ties	0		
	Total	12		

Comparison	R^+	R^-	p-value
SLAVE vs. Chi-RW	69.0	9.0	0.019607

Analysing the results showed in Table II, III and IV we can highlight that, although SLAVE algorithm does not obtain the best training accuracy in all the data sets, the test accuracy is the best in 11 of the 12 data sets and the models obtained have less number of rules. Moreover, the statistical analysis (pairwise comparisons Wilcoxon's test) obtaining that SLAVE algorithm clearly outperforms the Chi-RW algorithm assuming a high level of significance $p = 0.0196$.

VI. CONCLUSIONS

In this work, we have described KEEL, a software tool to assess EAs for DM problems, paying special attention to the GFS algorithms integrated in the tool. It relieves researchers of much technical work and allows them to focus on the analysis of their new GFS algorithms in comparison with the existing ones. Moreover, the tool enables researchers with a basic knowledge of fuzzy logic and evolutionary computation to apply GFSs to their work.

We have shown a case study to illustrate functionalities and the experiment set up processes in KEEL. In this case,

the results have been contrasted through statistical analysis (pairwise comparisons Wilcoxon's test), obtaining that SLAVE algorithm clearly outperforms the Chi-RW algorithm assuming a high level of significance $p = 0.0196$.

The KEEL software tool is being continuously updated and improved. At the moment, we are developing a new set of GFSs and a test tool that will allow us to apply parametric and non-parametric tests on any set of data. We are also developing data visualization tools for the on-line and off-line modules. We are also working on the development of a data set repository that includes the data set partitions and algorithm results on these data sets, the *KEEL-dataset*[5].

VII. ACKNOWLEDGMENTS
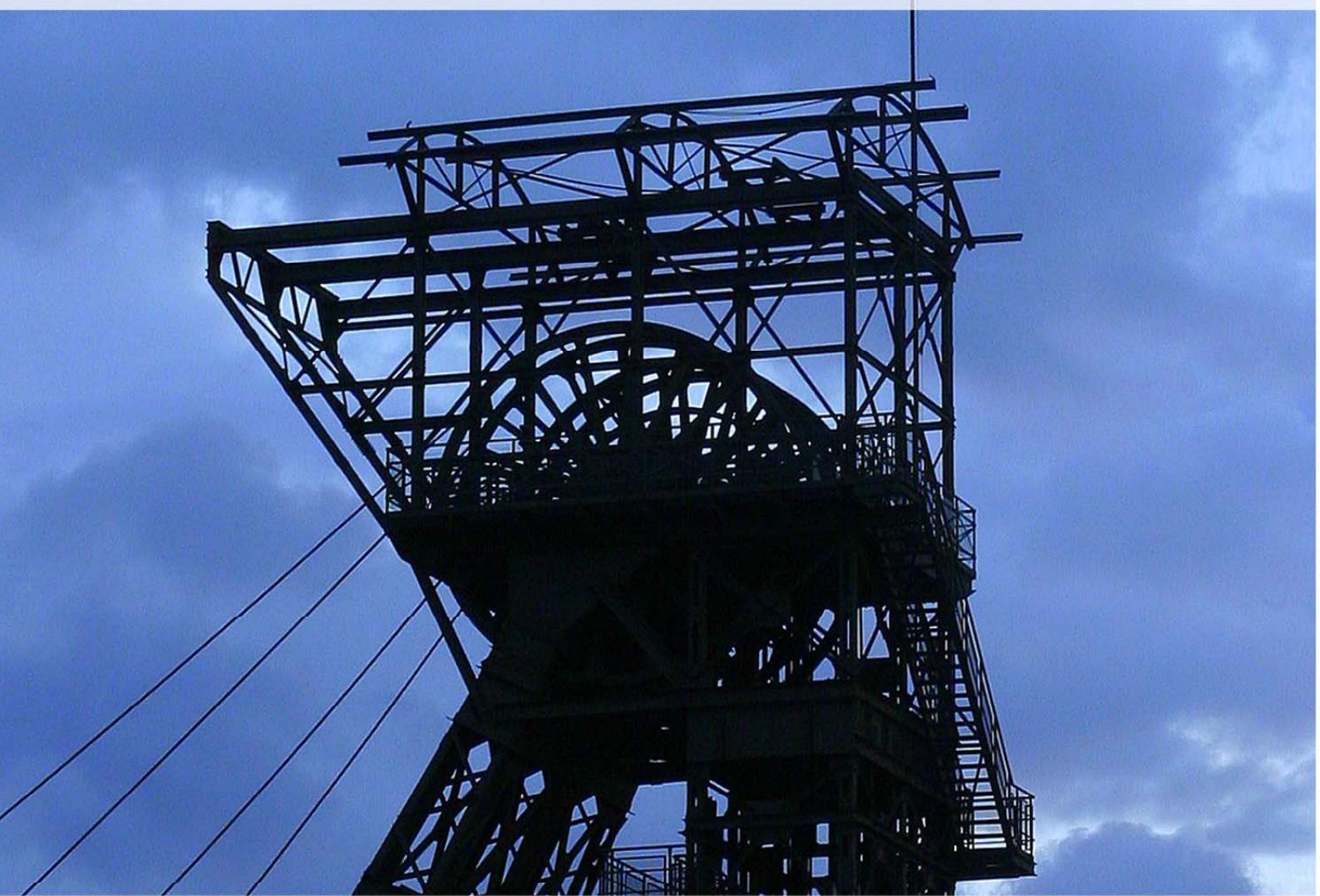
REFERENCES

[1] R.R. Yager and D.P. Filev, *Essentials of Fuzzy Modelling and Control*, John Wiley & Sons, USA (1994).

[2] D.E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley Professional, Canada (1989).

[3] J.H. Holland, *Adaptation in natural and artificial systems*, Ann Arbor: University of Michigan Press (1975).

[4] O. Cordón, F. Herrera, F. Hoffmann and L. Magdalena, *Genetic fuzzy systems. Evolutionary tuning and learning of fuzzy knowledge bases*, World Scientific, Singapore (2001).

[5] A.E. Eiben and J.E. Smith, *Introduction to Evolutionary Computing*, Springer-Verlag, Berlin (2003).

[6] R. Alcalá, J. Casillas, O. Cordón, A. González and F. Herrera, A Genetic Rule Weighting and Selection Process for Fuzzy Control of Heating, Ventilating and Air Conditioning Systems, *Engineering Applications of Artificial Intelligence* 18:3 (2005) 279-296.

[7] E. Van Broekhoven, V. Adriaenssens and B. De Baets, Interpretability-preserving genetic optimization of linguistic terms in fuzzy models for fuzzy ordered classification: An ecological case study, *International Journal of Approximate Reasoning* 44:1 (2007) 65-90.

[8] J.S. Shieh, M.H. Kao and C.C. Liu, Genetic fuzzy modelling and control of bispectral index (BIS) for general intravenous anaesthesia, *Medical Engineering and Physics* 28:2 (2006) 134-148.

[9] M. Mucientes, D.L. Moreno, A. Bugarín and S. Barro, Evolutionary learning of a fuzzy controller for wallfollowing behavior in mobile robotics, *Soft Computing* 4:10 (2006) 881-889.

[10] S. Sonnenburg, M.L. Braun, Ch.S. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K-R. Müller, F. Pereira, C.E. Rasmussen, G. Rätsch, B. Schölkopf, A. Smola, P. Vincent, J. Weston and R.C. Williamson, The Need for Open Source Software in Machine Learning, *Journal of Machine Learning Research* 8 (2007) 2443-2466.

[11] S. Smith, *A learning system based on genetic algorithms*, Ph.D. thesis, Unversity of Pittsburgh (1980).

[12] K.A. De Jong, W.M. Spears and D.F. Gordon, Using genetic algorithms for concept learning, *Machine Learning* 13 (1993) 161-188.

[13] J.H. Holland and J.S. Reitman, Cognitive systems based on adaptive algorithms, *SIGART Bull* 63 (1977) 49-49.

[14] T. Kovacs, *Strength or accuracy: Credit assignment in Learning Classifier Systems*, Springer-Verlag, Berlin (2004).

[15] G. Venturini, "SIA: a supervised inductive algorithm with genetic search for learning attribute based concepts", *in European Conference on Machine Learning*, LNCS Vol. 669 (1993) 280-296.

[16] D.P. Greene and S.F. Smith, Competition-based induction of decision models from examples, *Machine Learning* 3 (1993) 229-257.

[17] M.L. Wong and K.S. Leung, *Data mining using grammar based genetic programming and applications*, Kluwer Academic Publishers, Norwell (2000).

[18] J. Bacardit and J.M. Garrell, "Bloat control and generalization pressure using the minimum description length principle for a pittsburgh approach learning classifier system", *in Advances at the frontier of Learning Classifier Systems*, LNCS Vol. 4399 (2006) 61-80.

[19] S.W. Wilson, Classifier Fitness Based on Accuracy, *Evolutionary Computation* 3:2 (1995) 149-175.

[20] E. Bernadó-Mansilla and J.M. Garrell, Accuracy-Based Learning Classifier Systems: Models, Analysis and Applications to Classification Tasks, *Evolutionary Computation* 11:3 (2003) 209-238.

[21] J.S. Aguilar-Ruiz, R. Giráldez and J.C. Riquelme, Natural Encoding for Evolutionary Supervised Learning, *IEEE Transactions on Evolutionary Computation* 11:4 (2007) 466-479.

[22] F. Herrera, Genetic Fuzzy Systems: Taxonomy, Current Research Trends and Prospects, *Evolutionary Intelligence*, 2008. In press.

[23] F. Herrera, M. Lozano and J.L. Verdegay, Fuzzy Logic Controllers by Genetic Algorithms, *International Journal of Approximate Reasoning* 12 (1995) 299-315.

[24] R. Alcalá, O. Cordón and F. Herrera, "Combining Rule Weight Learning and Rule Selection to Obtain Simpler and More Accurate Linguistic Fuzzy Models", *in Modelling with Words*, LNCS Vol. 2873 (2003) 44-63.

[25] P. Thrift, Fuzzy logic synthesis with genetic algorithms, *in 4th International Conference on Genetic Algorithms (ICGA91)* (1991) 509-513.

[26] A. González and R. Perez, SLAVE: A genetic learning system based on an iterative approach, *IEEE Transactions on Fuzzy Systems* 7:2 (1999) 176-191.

[27] J. Casillas, O. Cordón and F. Herrera. COR: A methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules, *IEEE Transactions on System, Man and Cybernetics, Part B: Cybernetics* 32:4 (2002) 526-537.

[28] H. Ishibuchi, T. Yamamoto and T. Nakashima, Hybridization of Fuzzy GBML Approaches for Pattern Classification Problems, *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics* 35:2 (2005) 359-365.

[29] O. Cordón and F. Herrera, A Two-Stage Evolutionary Process for Designing TSK Fuzzy Rule-Based Systems, *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics* 29:6 (1999) 703-715.

[30] R. Alcalá, J. Alcala-Fdez, J. Casillas, O. Cordón and F. Herrera, Local Identification of Prototypes for Genetic Learning of Accurate TSK Fuzzy Rule-Based Systems, *International Journal of Intelligent Systems* 22:9 (2007) 909-941.

[31] O. Cordón and F. Herrera, Hybridizing Genetic Algorithms with Sharing Scheme and Evolution Strategies for Designing Approximate Fuzzy Rule-Based Systems, *Fuzzy Sets and Systems* 118:2 (2001) 235-255.

[32] H. Ishibuchi, K. Nozaki, N. Yamamoto and H. Tanaka, Selecting Fuzzzy If-Then Rules for Classification Problems Using Genetic Algorithms, *IEEE Transactions on Fuzzy Systems* 3:3 (1995) 260-270.

[33] S. Ventura, C. Romero, A. Zafra, J.A. Delgado and C. Hervás, JCLEC: A Java framework for Evolutionary Computation, *Soft Computing* 12:4 (2008) 381-392.

[34] O. Cordón, M.J. del Jesus, F. Herrera and M. Lozano, MOGUL: A Methodology to Obtain Genetic fuzzy rule-based systems Under the iterative rule Learning approach, *International Journal of Intelligent Systems* 14:11 (1999) 1123-1153.

[35] Z. Chi, J. Wu and H. Yan, Handwritten Numeral Recognition Using Self-Organizing Maps and Fuzzy Rules, *Pattern Recognition* 28:1 (1995) 59-66.

[36] H. Ishibuchi and T. Yamamoto, Rule weight specification in fuzzy rule-based classification systems, *IEEE Transactions on Fuzzy Systems* 13:4 (2005) 428-435.

[37] L.X. Wang and J.M. Mendel, Generating Fuzzy Rules by Learning from Examples, *IEEE Transactions on Systems, Man and Cybernetics* 22:6 (1992) 1414-1427.

[38] J. Demšar, Statistical Comparisons of Classifiers over Multiple Data Sets, *Journal of Machine Learning Research* 7 (2006) 1-30.

[5]http://www.keel.es/datasets.php

2008 3rd International Workshop on Genetic and Evolving Fuzzy Systems

Witten-Bommerholz, Germany
March 4-7, 2008

GEFS

IEEE Computational Intelligence Society

Regelungs-System-Technik

◈ IEEE

INTERNATIONAL FUZZY SYSTEMS ASSOCIATION

Sonder Forschungs Bereich

EUSFLAT

EUROPEAN SOCIETY FOR FUZZY LOGIC AND TECHNOLOGY

Session IV: Genetic Fuzzy Systems I

Session V: Genetic Fuzzy Systems II

Session VI: Genetic Programming-based Genetic Fuzzy Systems