FuzzySRI-II: A Fuzzy Rule Induction Algorithm for Numerical Output Prediction

Ashraf A. Afify

***Abstract*—Current inductive learning algorithms have difficulties handling attributes with numerical output values. This paper presents FuzzySRI-II, a new fuzzy rule induction algorithm for the prediction of numerical outputs. FuzzySRI-II integrates the comprehensibility and ease of application of rule induction algorithms with the uncertainty handling and approximate reasoning capabilities of fuzzy sets. The performance of the proposed FuzzySRI-II algorithm in two simulated control applications involving numerical output values is demonstrated and compared to that of the recently developed RULES-F Plus fuzzy rule induction algorithm. Results show that the rules derived from FuzzySRI-II are simpler and yield higher accuracy than those from RULES-F Plus.**

***Index Terms*—Fuzzy systems, rule induction, inductive learning, numerical output prediction, control systems**

I. Introduction

MANY engineering applications require the creation of models for the prediction of attributes with numerical values. Techniques have been developed to build approximate models for such a purpose. One of the most popular methods is the generation of models in the form of neural networks. This requires effort in determining the network configuration and produces "black box" type models that are difficult to interpret.

A number of studies have been conducted on the adaptation of typical inductive learning algorithms for the handling of numerical outputs. Many researchers have attempted to use decision-tree induction and have handled numerical output problems by simply dividing the output range into small intervals that can then be regarded as nominal class values. However, these attempts often fail [1] because they yield extremely large or inaccurate models.

Successful attempts have tended to employ regression tree algorithms, the most famous of which perhaps is CART [2]. A regression tree model is generated and interpreted in a similar way to a typical classification tree. The main differences appear in the interpretation of a leaf and in the splitting criterion employed. In CART, for a numerical attribute A^i, each internal node of the tree is a test of the type $[A^i > t^i]$, where t^i is a test value for A^i. At a particular leaf, the predicted output value is the mean of the output values of the instances in the training set T reaching the leaf. The splitting process is performed in order to minimise the mean squared error at each leaf. Based on the CART approach, many other algorithms have been developed. For instance, [3] presented a more accurate algorithm for the construction of region-splitting regression trees, similar to CART but including a wider range of possible tests for the internal nodes. With another algorithm, [4] employed a Bayesian approach to estimate the class distribution in tree-structured regression, which resulted in the modification of the splitting and pruning criterion and permitted the creation of trees with smaller classification errors.

Nevertheless, a problem with these types of regression trees is that they all employ constant values at their leaves. This restricts the use of the regression tree model when handling prediction problems because the number of possible predicted values will be limited by the number of leaves in the tree. Other attempts were based on rule induction algorithms, for example CN4 and KEX [5]. However, as in the original regression trees, the value predicted by each rule (equivalent to a leaf) is a single value.

To resolve this problem, a new type of regression method was developed. The method is based on the typical decision tree structure but it uses local linear regression functions instead of single values at the leaves [6]. A similar idea is adopted in the M5 algorithm [1], where at each leaf, the output prediction depends on a linear function with the attribute values as parameters. However, with the use of linear regression functions, inductive learning algorithms lose the characteristic of being close to human reasoning - the very characteristic that gave them their popularity.

A particular type of model, based upon fuzzy logic, has been adopted increasingly frequently for the development of expert systems and intelligent controllers, due to its similarity to some aspects of human reasoning [7], [8]. A key feature of fuzzy logic models is that they can handle vagueness and uncertainty. They can also deal with numerical outputs and do not require sophisticated mathematics. Fuzzy logic becomes useful when processing systems that are too complex for conventional methods or when the available information is qualitative, inexact or uncertain. A handful of methods have been proposed for automatic fuzzy rules generation [9]–[17]. However, many of these methods have been developed only for classification problems. Furthermore, for many of these methods the membership functions need to be predefined and this could be a difficult task. In fact, the problem of designing the membership function may be just as complex as designing

Manuscript received March 10, 2014; revised April 06, 2014.

A. A. Afify is with the Industrial Engineering Department, College of Engineering, King Saud University, Riyadh, SA, on leave from the Industrial Engineering Department, Faculty of Engineering, Zagazig University, Egypt, where he holds an Associate Professor position (phone: +966-1467-0669; fax: +966-1467-6875; e-mail: ash_afify@yahoo.com).

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

fuzzy rules.

Reference [18] introduced a fuzzy rule induction algorithm for numerical output handling called RULES-F Plus, which is based on the structure of the RULES-5 Plus crisp rule learner [19]. One of the main problems of RULES-F Plus is that it learns a complete and consistent rule set that tries to cover all of the training instances. In the case of noisy data, specific rules based on few training instances tend to be selected. Although these rules perform perfectly on the training instances, their predictive accuracy on future test instances is often lower because rules formed on the basis of small numbers of instances are susceptible to noise. Consequently, the rule set is both large and not of the highest accuracy.

This paper presents a new fuzzy rule induction algorithm called FuzzySRI-II (for Fuzzy Scalable Rule Induction - Version II) that enables the prediction of numerical output. FuzzySRI-II is based on the FuzzySRI classification learning algorithm [17], previously developed by the author. FuzzySRI-II employs a fast and noise-tolerant search method for extracting accurate and compact fuzzy If-Then rules from instances. One of its advantages compared with most existing fuzzy induction algorithms is that it integrates the fuzzy set techniques into the rule induction process and allows the automatic creation of membership functions.

The paper is organised as follows. For convenient reference, an outline of FuzzySRI is provided in Section II. Section III describes the proposed FuzzySRI-II algorithm. Experimental results are given in Section IV. Section V concludes the paper and provides suggestions for future work.

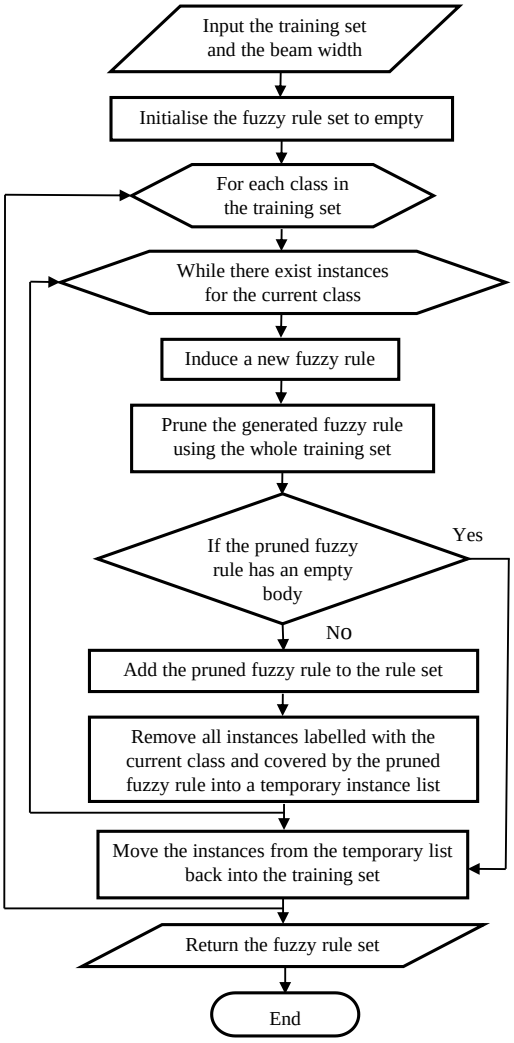

Fig. 1. A simplified description of the FuzzySRI algorithm.

II. The FuzzySRI Algorithm

FuzzySRI (for Fuzzy Scalable Rule Induction) induces fuzzy classification rules from a training set of instances with known nominal classes. Each instance in the training set is represented by a vector of attributes called linguistic variables. Each linguistic variable is described by a set of linguistic values called linguistic terms. An attribute is either nominal or numerical. In FuzzySRI, each fuzzy rule, or fuzzy concept description, consists of a conjunction of antecedents and a predicted nominal class. Each antecedent is a fuzzy condition on a single attribute and there is at most one antecedent per attribute.

FuzzySRI follows the general one-rule-at-a-time procedure of rule induction algorithms. It searches the rule space in a top-down fashion. Fig. 1 provides a simplified description of FuzzySRI. The algorithm starts with an empty fuzzy rule set. Given a training instance list and a beam width, it generates fuzzy rules for each class in turn. Having chosen a class on which to focus, it extracts a new fuzzy rule that will cover a subset of the positive instances (instances belonging to the chosen class). The generated fuzzy rule is then simplified using an incremental pruning procedure based on the minimum description length (MDL) principle [20]. If the pruned fuzzy rule has an empty body, it is assumed that no further rule can be found that explains the remaining positive instances and the learning process stops for the current class. Otherwise, the pruned fuzzy rule is added to the fuzzy rule set and all covered positive instances are temporarily removed from the training set. Fuzzy rules are learned in this way until no positive instances are left. Once all the fuzzy rules for one class are produced, all removed instances are put back into the training set before inducing fuzzy rules for the second class. In this way, the algorithm is always based on all available training instances to form fuzzy rules for a specific class. This whole process is repeated for each class in turn to produce an unordered set of fuzzy rules.

To generate a fuzzy rule, FuzzySRI performs a pruned general-to-specific beam search for a fuzzy rule that optimises a given quality criterion. The search for fuzzy rules aims to cover as many positive instances and as few negative instances as possible (negative instances are those not belonging to the class under consideration). FuzzySRI uses several effective search-space pruning heuristics to avoid useless specialisations and to terminate a non-productive search during rule construction, which substantially increases the efficiency of the learning process. A detailed description of the learning of single fuzzy rules procedure can be found in [17].

The FuzzySRI algorithm uses an intuitive way to obtain fuzzy intervals for numerical attributes. First, it discretises

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

the domain of each attribute into several crisp intervals using an off-line entropy-based discretisation method [21]. Second, it derives fuzzy intervals from the crisp ones by defining membership functions on the domains of the attributes. The derived intervals are then treated in a similar way to nominal values during induction.

III. The FuzzySRI-II Algorithm

FuzzySRI-II is an extension of the FuzzySRI classification rule learner [17]. This section describes modifications to the FuzzySRI algorithm to allow the prediction of numerical outputs.

A. Representation

FuzzySRI-II creates fuzzy If-Then rules directly from a set of instances called the training set *T*. Each instance *I* is described by a vector of n_a input attributes ($A^1, .., A^i, ..., A^{n_a}$) and an output attribute A^o. The input (v_I^i) and output (v_I^o) attribute values in instance *I* are either nominal or numerical. An instance *I* can therefore be formally defined as follows:

$$I = (A^1 = v_I^1, \ldots, A^i = v_I^i, ..., A^{n_a} = v_I^{n_a}, A^o = v_I^o) \quad (1)$$

A fuzzy rule *R* can be written in the form:

$$\text{If } (A^1 \text{ is } L_R^1) \wedge ... \wedge (A^i \text{ is } L_R^i) \wedge ... \wedge (A^{n_a} \text{ is } L_R^{n_a}) \text{ Then } (A^o \text{ is } L_R^o) \quad (2)$$

where L_R^i and L_R^o are respectively the linguistic values (terms) of the i^{th} input attributes and the output attribute in rule *R*. Each linguistic value can be obtained by defining membership functions on the domains of the attributes. Typical shapes of membership functions are triangular, trapezoidal, and bell-shaped. FuzzySRI-II adopts triangular forms in this study as they are simple and often used in fuzzy sets. A triangular membership function can be defined as *Tr*(*a*,*b*,*c*), where *ac* being the base of the triangle and *b* the location of its apex (Fig. 2). It should be noted that the values of nominal attributes can be converted to linguistic values by assigning crisp functions with membership degrees of either 0 or 1. A rule set indicates the disjunction of a number of rules $\{R_1, ..., R_l, ..., R_{n_r}\}$, and it is denoted as $RS = \{R_1 \vee ... \vee R_l \vee ... \vee R_{n_r}\}$.

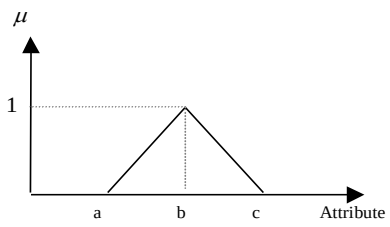

Fig. 2. Triangular fuzzy set.

B. Formation of a Fuzzy Rule

The FuzzySRI rule forming process has been designed for instances with nominal classes. Therefore, in order to be able to use this process, the numerical output values of all instances in the training set *T* need to be fuzzified. It is proposed to split the numerical output range of each instance ($v_{\min}^o, v_{\max}^o$), where $v_{\min}^o$ is the minimum known value for the output attribute and $v_{\max}^o$ its maximum known value, into a fixed number (n_l) of linguistic values. n_l is determined by the user and can be seen as a precision degree prescribed for the model; the higher the number, the higher the accuracy of the created rule set. However, this will also cause the number of rules to increase. Thus, the user has a degree of control over the size and precision of the model.

Given the numerical output attribute range and the number n_l of required linguistic values, the FuzzySRI-II algorithm decomposes the output range into n_l triangular linguistic values (L_1^o, ... , L_k^o, ... , $L_{n_l}^o$) defined as: L_k^o = *Tr*(*a*(*k*),*b*(*k*),*c*(*k*)), where *k* is an integer included in [1, n_l], $b(k) = [(v_{\max}^o - v_{\min}^o)/(n_l - 1)].(k-1)$, $a(k) = b(k) - (v_{\max}^o - v_{\min}^o)/(n_l - 1)$, and $c(k) = b(k) + (v_{\max}^o - v_{\min}^o)/(n_l - 1)$. For instance, Fig. 3 shows the fuzzification when $v_{\min}^o = 0$, $v_{\max}^o = 15$ and $n_l = 4$.

Based on this decomposition, there could be two possible linguistic values with respect to which the output attribute value (v_I^o) in a given instance *I* can be assigned. The output linguistic value will be the value L_k^o where the membership degree $\mu_{L_k^o}(v_I^o)$ is maximum, $v_I^o \in \left](b(k) - a(k))/2, (c(k) - b(k))/2\right[$, and $\mu_{L_k^o}(v_L^o) > 0.5$. In the particular case where the membership degree of v_I^o is equal for two linguistic values (L_k^o and L_{k+1}^o), i.e. $v_I^o = (b(k+1) - a(k+1))/2 = (c(k) - b(k))/2$ and $\mu_{L_k^o}(v_L^o) = 0.5$, one is selected randomly.

Now that the output linguistic values are defined, the FuzzySRI-II algorithm can select an instance with its corresponding output linguistic value in order to form a fuzzy rule. The rule forming process of FuzzySRI is then used unchanged. At the end of the rule formation process, a

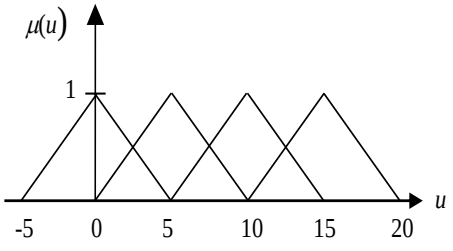

Fig. 3. Output fuzzification.

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

rule R is obtained to cover as many positive instances and as few negative instances as possible.

C. Output Prediction

FuzzySRI-II adopts the following defuzzification strategy to predict the output value of a particular instance I. First, the membership degree of the instance I with each rule R, $\mu_R(I)$, is computed, namely:

$$\mu_R(I) = \prod_{i=1}^{n_a} (\mu_{L_R^i}(v_I^i)) \quad (3)$$

where $\mu_{L_R^i}(v_I^i)$ is the membership degree of each attribute value (v_I^i) in the instance I with regard to the corresponding linguistic value L_R^i in the rule R. Then, the weighted average method [22] is used to compute the defuzzified output:

$$output = \sum_{l=1}^{n_r} \mu_{R_l}(I) \cdot C_{out} / \sum_{l=1}^{n_r} \mu_{R_l}(I) \quad (4)$$

where I is the new instance, C_{out} the centre of the output fuzzy set of the rule being considered and n_r the total number of rules.

IV. Tests and Analysis of Results

A comparison between FuzzySRI-II and RULES-F Plus algorithms has been carried out. Two practical cases have been used. The first problem aims to develop a model for the guidance of a robotic arm and the second one concerns the control of a truck. These two problems were employed to evaluate the RULES-F Plus algorithm [18]. Three measures were used to assess the performance of each created model, namely, the number of created rules, the maximum absolute error (Maximum E_{abs}) and the mean absolute error (Mean E_{abs}) between the real output values and the values predicted by the models obtained with the instances in T.

A. Fuzzy Model for Robot Arm Control

The problem involved the creation of a fuzzy model for the control of a PUMA 560 robot [23]. The position (X, Y, Z) of the robot end effector depends on three joints and can be defined by the angles $\theta1$, $\theta2$, $\theta3$ at these joints. The training set T contains 27,825 instances. An instance in T is composed of six input attributes (the joint angles $\theta1_t$, $\theta2_t$, and $\theta3_t$ at time t and the joint angles $\theta1_{t-1}$, $\theta2_{t-1}$, and $\theta3_{t-1}$ at time t-1) and three outputs (the resulting spatial positions X_{t+1}, Y_{t+1}, Z_{t+1}).

Models were created using RULES-F Plus and FuzzySRI-II. For both algorithms, the universes of discourse of the three outputs were decomposed into ten linguistic values. In addition, because RULES-F Plus and FuzzySRI-II can only handle one output at a time, three training sets were generated in order to produce one rule set for each output. Finally, RULES-F Plus and FuzzySRI-II each has a number of parameters whose values determine the quality of their induced fuzzy rule sets. The experiments reported in [18] set the *PRSET_size* to 2 and no pruning process was employed. Therefore, these parameter values were used in this study. The default parameters of FuzzySRI were used [17].

Results are shown in Table I. Considering all performance measures, FuzzySRI-II clearly outperforms the RULES-F Plus algorithm. FuzzySRI-II produced a much smaller model than that created by RULES-F Plus, with smaller values for the Maximum E_{abs} and Mean E_{abs} measures. To illustrate its performance, the predictions of the model created by FuzzySRI-II for the first 10000 instances compared with the actual outputs X, Y, and Z are shown in Figs. 4, 5, and 6 respectively.

B. Fuzzy Model for Truck Control

The aim of this experiment was to create a model for the control of a truck reversing to a specified loading dock. Each instance in T represents one position of the truck (Fig. 7), defined by the angle φ of the truck relative to the horizontal and the location x of the truck on the horizontal axis, with the resulting required action on the steering wheel, defined by the required steering angle θ. Thus, an instance is composed of 2 input attributes φ and x and one output θ. T contains 238 instances.

For both RULES-F Plus and FuzzySRI-II, the output universe of discourse was decomposed into seven linguistic values. RULES-F Plus was applied twice, once with and once without the IPP pruning process [19]. For both cases, *PRSET_size* was fixed at a value of 2.

Results are shown in Table II. FuzzySRI-II once again clearly outperforms RULES-F Plus. When using the IPP pruning process (with noise level (*NL*) equals 0.25), RULES-F Plus created a more compact rule set but with a substantial reduction in accuracy. However, for the case illustrated, the rule set created by FuzzySRI-II was more compact and still more accurate than the model created by the RULES-F Plus algorithm.

TABLE I
FUZZY INDUCTION RESULTS FOR THE ROBOT ARM CONTROL PROBLEM

	Robot arm control											
	Combined results			X position			Y position			Z position		
	Total number of rules	Average maximum	Average mean	Number of rules	Maximum E_{abs}	Mean E_{abs}	Number of rules	Maximum E_{abs}	Mean E_{abs}	Number of rules	Maximum E_{abs}	Mean E_{abs}
RULES-F Plus	55	0.0754	0.0129	13	0.1025	0.0222	14	0.0889	0.0121	28	0.0438	0.0044
FuzzySRI-II	31	0.0404	0.006	7	0.0612	0.0103	8	0.0384	0.0058	16	0.0217	0.0019

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

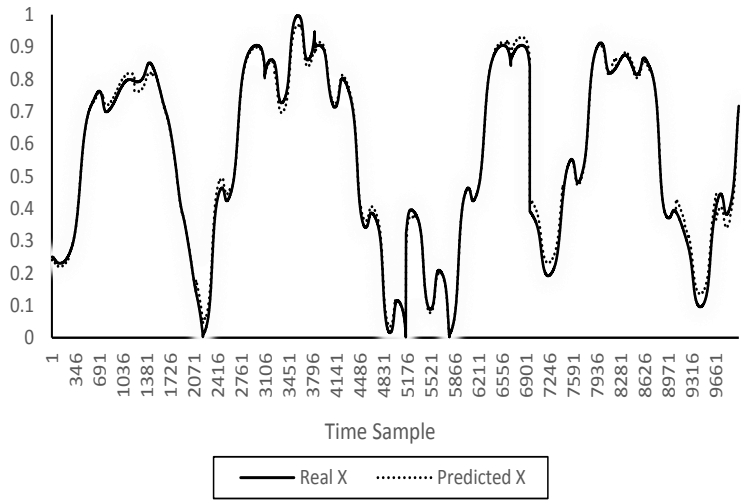

Fig. 4. FuzzySRI-II prediction for the output *X*.

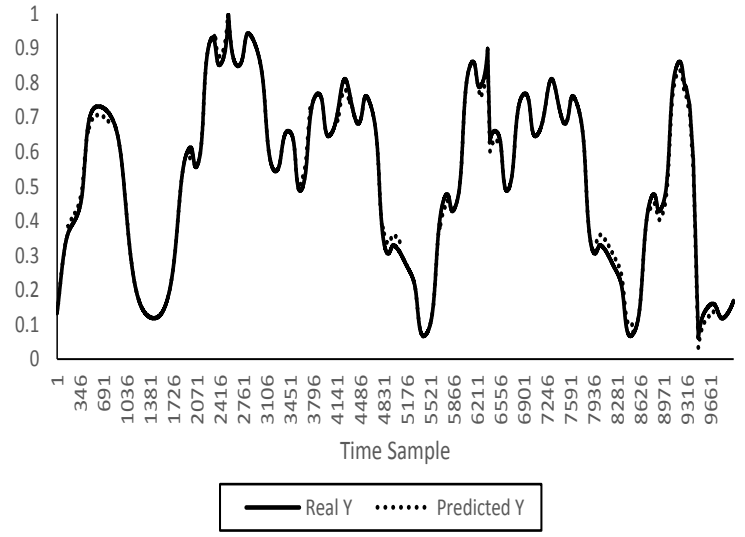

Fig. 5. FuzzySRI-II prediction for the output *Y*.

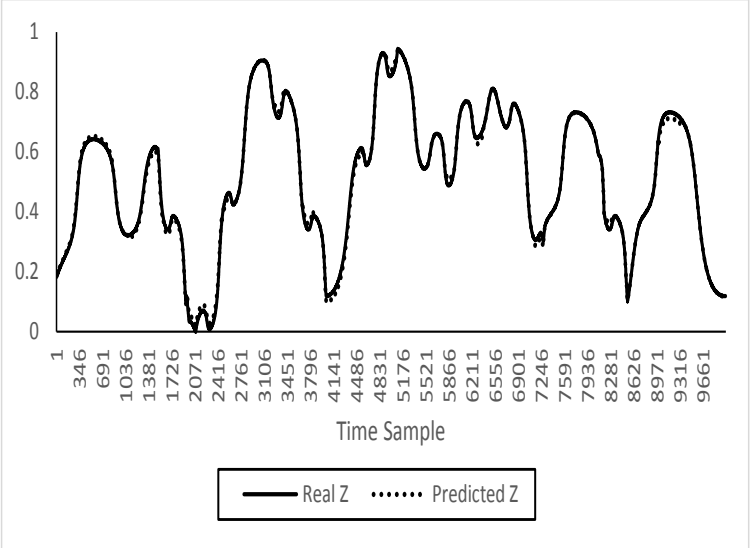

Fig. 6. FuzzySRI-II prediction for the output *Z*.

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

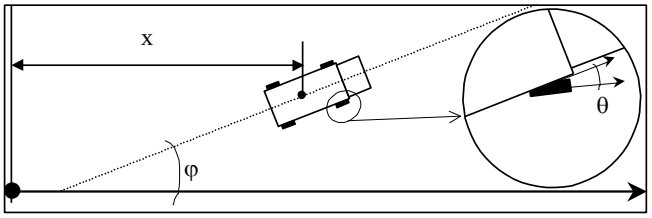

Fig. 7. Truck control problem.

TABLE II
FUZZY INDUCTION RESULTS FOR THE TRUCK CONTROL PROBLEM

	Truck control		
	Number of rules	Maximum E_{abs}	Mean Eabs
RULES-F Plus	46	11.4	3.46
RULES-F Plus, $NL = 0.25$	12	42.3	6.24
FuzzySRI-II	8	8.6	2.37

V. Conclusions and Future Work

This paper has presented a new method for fuzzy rule induction based on a modification of the FuzzySRI rule induction classifier to enable the creation of models for the prediction of numerical outputs. The resulting algorithm, called FuzzySRI-II, combines the capabilities of fuzzy logic for numerical output and uncertainty handling with the good performance of FuzzySRI. Tests on two control problems have shown that, compared to the recently developed RULES-F Plus fuzzy rule induction algorithm, FuzzySRI-II permits the creation of more compact and more accurate fuzzy rule sets. Thus, this research has yielded an algorithm that seems appropriate for control applications, which is a departure from the typical applications of inductive learning algorithms to date.

Further tests should be carried out to compare the new FuzzySRI-II algorithm with other well-established methods, such as neuro-fuzzy algorithms. Also, tests have shown that the performance of the algorithm is highly dependent on the number of output membership functions used. Further research could be conducted to automate the creation or refinement of output membership functions.

Acknowledgment

The author wishes to thank the Industrial Engineering Department at King Saud University, Saudi Arabia for providing a good environment, facilities and financial means to complete this paper.

References

[1] J. R. Quinlan, "Learning with continuous classes," in *Proc. 5th Australian Joint Conf. Artificial Intelligence*, Singapore, 1992, pp. 343–348.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Belmont, CA, USA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[3] Y. Morimoto, H. Ishii, and S. Morishita, "Efficient construction of regression trees with range and region splitting," in *Proc. 23rd Very Large Data Bases Conf.*, Athens, Greece, 1997, pp. 166–175.

[4] A. Karalic, "Employing linear regression in regression tree leaves," in *Proc. Euro. Conf. Artificial Intelligence*, Vienna, Austria, 1992, pp. 440–441.

[5] I. Bruha, and P. Berka, "Continuous classes in rule induction: Empirical comparison of two approaches," in *Proc. 3rd Int. Workshop Artificial Intelligence Techniques*, Brno, Czech Republic, 1996, pp. 163–164.

[6] A. Karalic, and B. Cestnik, "The Bayesian approach to tree-structured regression," in *Proc. Information Technology Interfaces*, Cavtat, Yugoslavia, 1991, pp. 155–160.

[7] G. J. Klir, and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, New Jersey, USA: Prentice-Hall, 1995.

[8] E. Cox, *The Fuzzy Systems Handbook: A Practitioner's Guide to Building, Using, and Maintaining Fuzzy Systems*. 2nd ed., London, UK: Academic Press, 1998.

[9] L. X. Wang, and J. M. Mendel, "Generating fuzzy rules from numerical data, with applications," Signal and Image Processing Institute, University of Southern California, CA, USA, TR USC-SIPI #169, 1991.

[10] D. Nauck, and R. Kruse, "NEFCLASS – a neuro-fuzzy approach for the classification of data," in *Proc. ACM Symposium Applied Computing*, Nashville, Tennessee, USA, 1995, pp. 461–465.

[11] Y. Yuan, and M. J. Shaw, "Induction of fuzzy decision trees," *Fuzzy Sets and Syst.*, vol. 69, no. 2, pp. 125–139, 1995.

[12] Y. Yuan, and H. Zhuang, "A genetic algorithm for generating fuzzy classification rules," *Fuzzy Sets and Syst.*, vol. 84, no. 1, pp. 1–19, 1996.

[13] B. Jeng, Y. M. Jeng, and T.P. Liang, "FILM: A fuzzy inductive learning method for automated knowledge acquisition," *Decision Support Syst.*, vol. 21, no. 2, pp. 61–74, 1997.

[14] H. P. Störr, "A compact fuzzy extension of the Naive Bayesian classification algorithm," in *Proc. 3rd Int. Conf. Intelligent Technologies and Vietnam-Japan Symposium Fuzzy Systems and Applications*, Hanoi, Vietnam, 2002, pp. 172–177.

[15] I. Cloete, and J. van Zyl, "Fuzzy rule induction in a set covering framework," *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 1, pp. 93–110, 2006.

[16] J. Hühn, and E. Hüllermeier, "FURIA: An algorithm for unordered fuzzy rule induction," *Data Mining and Knowledge Discovery*, vol. 19, pp. 293–319, 2009.

[17] A. A. Afify, "A novel algorithm for fuzzy rule induction in data mining," *Proc. Inst. Mech. Engs., Part C: J. Mech. Eng. Sc.*, vol. 228, no. 5, pp. 877–895, 2014.

[18] S. Bigot, "A new rule space representation scheme for rule induction in classification and control applications," *Proc. Inst. Mech. Engs., Part I: J. Syst. and Cont. Eng.*, vol. 225, no. 7, pp. 1018–1038, 2011.

[19] D. T. Pham, S. Bigot, and S. S. Dimov, "RULES-5: A rule induction algorithm for problems involving continuous attributes," *Proc. Inst. Mech. Engs., Part C: J. Mech. Eng. Sc.*, vol. 217, no. 12, pp. 1273–1286, 2003.

[20] D. T. Pham, and A. A. Afify, "Three new MDL-based pruning techniques for robust rule induction," *Proc. Inst. Mech. Engs., Part C: J. Mech. Eng. Sc.*, vol. 220, no. 4, pp. 553–564, 2005.

[21] U. M. Fayyad, and K. B. Irani, "Multi-interval Discretization of Continuous-valued Attributes for Classification," in *Proc. 13th Int. Joint Conf. Artificial Intelligence*, Chambery, France, 1993, pp. 1022–1027.

[22] L. Rondeau, R. Ruelas, L. Levrat, and M. Lamotte, "A defuzzification method respecting the fuzzification," *Fuzzy Sets and Syst.*, vol. 86, pp. 311–320, 1997.

[23] B. Armstrong, and O. Khatib, "The explicit dynamic model and inertial parameters of the PUMA 560 Robot arm," in *Proc. 1986 Int. Conf. Robotics and Automation*, San Francisco, USA, 1986, pp. 510–518.