A FRAMEWORK FOR EVOLVING FUZZY CLASSIFIER SYSTEMS USING GENETIC PROGRAMMING

Brian Carse and Anthony G. Pipe

Faculty of Engineering, University of the West of England, Bristol BS16 1QY, United Kingdom. Brian.Carse, Anthony.Pipe@uwe.ac.uk

Abstract

A fuzzy classifier system framework is proposed which employs a tree-based representation for fuzzy rule (classifier) antecedents and genetic programming for fuzzy rule discovery. Such a rule representation is employed because of the expressive power and generality it endows to individual rules. The framework proposes accuracy-based fitness for individual fuzzy classifiers and employs evolutionary competition between simultaneously matched classifiers. The evolutionary algorithm (GP) is therefore searching for compact fuzzy rule bases which are simultaneously general, accurate and co-adapted. Additional extensions to the proposed framework are suggested.

Introduction

The fusion of rule-based representations and evolutionary algorithms has been, and continues to be, the focus of much attention as a basis for computational learning systems. The Learning Classifier System (LCS), devised by Holland (Holland, 1988), is an early example of a learning system that employs artificial evolution to evolve rule sets for problem solving. Following on from Holland's pioneering work, the LCS has been developed, refined and extended in many directions. A particularly significant direction has been to incorporate fuzzy sets and fuzzy inference into the LCS framework. Such Learning Fuzzy Classifier Systems are able to deal with problems where variables are real-valued, rule-bases are difficult to design by humans, and where linguistic interpretability is desirable.

This contribution brings together and builds upon a number of recent ideas to propose a novel fuzzy classifier system framework which operates in the "Michigan" style and employs a tree-based rule representation with Genetic Programming (Koza, 1992) used as the rule discovery mechanism. Individual classifier fitness is based on accuracy rather than strength (as proposed in Wilson's discrete-valued XCS (Wilson, 1994)).

Related Work

Previous Work Using Genetic Algorithms To Evolve Fuzzy Rule Bases

A large amount of research has been carried out employing the genetic algorithm in determining fuzzy system parameters. This subsection briefly summarises this work; for a more detailed overview than space permits here please see (Carse, 1996) and (Bonarini, 2000). The book (Herrera and Verdegay, 1996) contains a large compendium of relevant works combining genetic and fuzzy approaches.

The first major distinction among extant approaches in genetic optimisation of fuzzy system parameters is the way in which the GA is applied. With the so-called "Michigan" approach, the individual, as far as the GA is concerned, is a single rule or classifier. During learning episodes, each rule accrues some form of strength or fitness through interaction with the environment. When the evolutionary algorithm is applied, competition is between individual rules based on fitness. An alternative approach, called the "Pittsburgh" approach, maintains a population of rule-sets: each individual as far as the GA is concerned is a complete assembly of rules encoded on an appropriate genotype. Complete rule sets accrue strength through interaction with the environment and genetic operators, such as selection, reproduction and recombination, apply to these whole rule sets (which may be of fixed or variable size, depending on the encoding). Clearly the role of the GA in the two approaches is different as are the known difficulties: in Michigan-style systems a careful balance must be set between cooperation and competition between individual rules; in Pittsburgh-style systems reinforcement bandwidth is usually smaller and genetic crossover can be a cause of disruption. Indicative works using the Michigan approach include (Bonarini, 1997, 2000), (Parodi and Bonelli, 1993), (Valenzuela-Rendon, 1991) and works using the Pittsburgh approach include (Carse et al. 1996, 1998), (Hwang and Thompson, 1994), (Thrift, 1993).

Further differences in existing work using GAs to optimise fuzzy systems arise from the fuzzy system parameters to which the GA is applied. Common approaches include using the GA to learn:

1. Fuzzy rule-bases only with fixed membership functions (Bonarini 1997,2000), (Glorennec 1996), (Valenzuela-Rendon, 1991),
2. Membership functions only with fixed fuzzy rule-bases (Karr, 1991).
3. Fuzzy rule-bases and membership functions in stages (Ke et al., 1997), (Kinzel et al., 1994), (Rahmoun and Benmohamed 1998).
4. Fuzzy rule-bases and membership functions simultaneously (Carse 1996, 1998), (Lee and Takagi, 1993), (Liska and Melsheimer 1994), (Tang et al., 1998).

In the vast majority of existing work, the genotypes which encode parameters to be optimised (rule-bases, membership functions) are structured as bit strings or vectors of real numbers to which standard genetic algorithm recombination operators such as crossover, mutation and inversion are applied. In many cases, the genotype representation restricts the rule syntax to forms similar to

IF (x_1 is A_1) AND (x_2 is A_2) .. AND (x_k is A_k) THEN
(y_1 is B_1), (y_2 is B_2) .. (y_j is B_j)

where x_i are input variables, A_i are input membership functions, y_i are output variables and B_i are output membership functions. These representations effectively impose a grid partitioning on the input space and the number of rules can become very large as the number of inputs increases. Such genotype codings are often supplemented with a "don't care" label which indicates whether or not a particular part of the rule antecedent (or consequent) is inactive. Such "don't cares" allow the representation of more general rules.

An alternative approach is to represent individual rules as tree expressions using an appropriate set of logical function nodes such as AND, NOT and OR. This is the representation proposed in the current fuzzy classifier system framework. Such a representation naturally allows rules ranging from the most simple (and general) to relatively complex, as appropriate to the requirements of a high performance rule base. Since the technique of Genetic Programming (GP) is designed to operate on such tree structures, it is the evolutionary search mechanism proposed here. The next subsection briefly summarises previous work carried out using GP to evolve rule bases.

Previous Work Using Genetic Programming To Evolve Rule Bases

Genetic programming has been employed as the search mechanism in a number of rule-base learning studies, using both Michigan and Pittsburgh approaches applied to discrete-valued and fuzzy classifier systems. In (Edmunds et al. 1995) GP is applied to evolution of fuzzy rules with application to financial trading. This work applies GP in the Pittsburgh-approach in order to learn individual complex fuzzy rules to maximise investment profit. (Bastian 2000) applies GP to identify input variables, the rule base and involved membership functions for a test fuzzy model. Although the fuzzy rule base is successfully learned, it is reported that individual membership functions were not retrieved perfectly. (Akbarzadek et al. 2000) apply GP to learning of fuzzy navigational behaviour of a mobile robot. GP is used in the Pittsburgh approach to evolve individual rules for successful goal-seeking behaviour in a complex environment. (Bentley 1999, 2000) employs GP for evolving fuzzy rules for pattern classification and fraud detection. As a first stage, clustering is used to determine the domains and membership functions of input fuzzy sets. GP is then applied (Pittsburgh approach), using a binary encoded genotype and modified recombination operators. The results reported in these works indicate that a GP approach to learning fuzzy rules is a promising area for further investigation.

GP has also been applied to learning in Michigan-style discrete valued classifier systems. In (Lanzi and Perrucci, 1999), the XCS classifier system (Wilson 1995) is extended to represent rule antecedents as LISP s-expressions on which the GP operates. Experimental results are provided which demonstrate the efficacy of the approach to the multiplexor problem and to a multi-step environment learning problem. In (Ahluwalia and Bull, 1999), s-expressions are used as rule consequents (actions) in a Michigan-style discrete-valued classifier system applied to letter image recognition and credit card classification problems.

Some Key Issues in Michigan-style Classifier Systems

This section describes some key issues which influence the design of the Michigan-style Fuzzy Classifier System (MFCS) proposed. These issues are based on recent and current research into discrete valued MCS and are extended and discussed in the fuzzy case. Although these issues have been separated out here for presentation purposes, it should be stated that they are strongly inter-related. Note also that a further important key issue - that of credit assignment to individual classifiers - is not discussed in detail here.

Generalised Rules

In a discrete-valued classifier system, generalised rules are obtained by using '#' symbols ("don't cares') in the classifier syntax. The '#' symbol matches both '0' and '1' so, for example, the classifier condition 11## matches the input messages 1100, 1101, 1110 and 1111. Such general rule antecedents also arise naturally in the fuzzy case. For example, for a two input fuzzy system (with inputs x0 and x1) the rules:

IF (x0=NS) THEN..
IF (x0=NS) OR (x1=PL) THEN..
IF NOT((x0=NS) AND (x1=Z)) THEN..

are, to differing extents, general rules. Of course, the representation of such general rules, so long as they provide high reward outputs over the complete range of inputs, is potentially a very powerful one and allows the learning system to economically capture generalisations in the problem input/output mapping. This results in many fewer rules than would be required, for example, using a "grid-based" rule base.

However, such a generalisation capability has been known for a long time to provide problems for discrete valued MCSs and this also applies to the fuzzy case. The main problem is that of proliferation of "overgeneral" rules: rules which match many input states but whose outputs are only correct for a subset of input states and are incorrect for others. Despite being unreliable, such overgeneral rules can have more influence and better chances of survival (under action of the EA) than other more specific and correct rules with which they compete. One approach in overcoming this problem is to base rule fitness on accuracy rather than accrued strength from environmental reward. This is discussed next.

Strength Based versus Accuracy Based Classifier Fitness

Traditional Michigan-style classifier systems have been "strength-based" in the sense that a classifier accrues strength during interaction with the environment (through rewards and/or penalties). This strength is then used for two purposes: resolving conflicts between simultaneously matched classifiers during learning episodes; and as the basis of fitness for the evolutionary algorithm. A number of problems arise from this dual use of classifier strength. These include:

1. The cooperation/competition problem briefly discussed above. High-strength, potentially cooperative classifiers go on to compete under the action of the evolutionary algorithm.

2. Over-general rules with relatively high (but inconsistent) payoff can come to dominate the population.

3. In some environmental states, the maximum payoff achievable (by performing the best possible action for that state) may be relatively low. Although a classifier might be the best that can exist for that state, it can be eradicated from the population by other classifiers which achieve higher rewards in other states. This results in gaps in the system's "covering map".

In (Wilson, 1995) a completely different approach is taken in which a classifier's fitness, from the point of view of the evolutionary algorithm, is based on its "accuracy" i.e. how well a classifier predicts payoff whenever it fires. Classifier strength is still used for resolving conflicts between simultaneously firing classifiers. Such an accuracy based approach offers a number of advantages. Firstly, it can distinguish between accurate and overgeneral classifiers: an overgeneral classifier will have relatively low accuracy since payoff will vary according to the input states covered by the classifier. Indeed, it has been shown that the accuracy based approach can lead to evolution of optimally general classifiers (Kovacs, 1997). Additionally it can maintain both consistently correct and consistently incorrect classifiers which allows learning of a complete "covering map". A potential drawback of the accuracy based approach is that it is likely to require larger populations of classifiers, although its better generalisation capabilities may offset this to some extent.

SubPopulations to which EA is Applied

In a classifier system, the evolutionary algorithm is commonly employed as the discovery component. It has been observed that in a Michigan-style classifier system, the evolutionary algorithm is faced with an implicitly multi-objective optimisation task. The classifier system is required to simultaneously evolve a collection of classifiers, each of which is optimised to solve part of the overall problem.

In early Michigan-style classifier systems, the complete set of classifiers forms the population on which selection, recombination and replacement operate. This clearly does not address the implicit multi-objective nature of the problem on which the evolutionary algorithm operates. In (Horn, Goldberg and Deb, 1994) a "niching" approach is employed and shows that high quality and diverse niches can be evolved successfully.

More recently, further restrictions have been placed on the sub-population on which the evolutionary algorithm operates. For example, in Wilson's XCS (Wilson, 1995), the populations on which the evolutionary algorithm are applied are match sets i.e. the set of classifiers which match a particular environment input message. A more extreme approach, in which the population for selection consists of classifiers with the same antecedent, is advocated in (Bonarini 1997) for a fuzzy classifier system. At the expense of potentially larger numbers of classifiers, these approaches address directly the multi-objective nature of the problem in that competition under action of the EA is between classifiers which match the same input state but provide different outputs.

Proposed Framework

Classifier Representation

Each classifier antecedent is represented as a tree with function nodes AND, OR and NOT and a leaf node consists of an input variable and an input fuzzy membership function. Figure 1 shows the tree representation for the classifier

IF (x_3 = PL) OR ((x_1=ML) AND (x_2=Z)) THEN y = PS

The classifier consequent is an output variable and an associated output membership function.

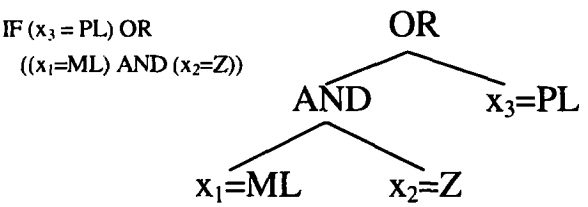

Figure 1. Tree-Based Rule Antecedent Representation

Evolutionary Algorithm and Operators

Since a tree-based rule representation is used, Genetic Programming (GP) is the natural rule discovery algorithm. Basic GP crossover selects random subtrees from two parents and exchanges these. Three mutation operators are used: an operator similar to standard GP mutation where a randomly selected subtree is replaced by a different randomly generated subtree and two operators where an input(output) membership function in a rule antecedent(consequent) is replaced by a different randomly selected input(output) membership function. Special care must be taken to ensure that these operators, and the initial random rule-base generator, do not create "nonsense" rule antecedents such as IF ((x1=NL) AND (NOT(x1=NL)). In addition to these standard operators, a cover operator is invoked if an input vector is encountered which no fuzzy rules match. The cover operator generates a random rule which matches the input vector with a minimum activation threshold, T. This random rule is then inserted into the population and activated to produce an output action.

Associated with each classifier, as in XCS, are a *strength, reward prediction* and *accuracy*. A classifier's strength is an estimate of the actual accrued reward received by the classifier. The reward prediction is an estimate of the expected reward when that classifier fires, and accuracy is a measure of how accurate that prediction is compared to the actual reward received. A classifier's reproductive fitness is proportional to the inverse of its accuracy. Classifiers for reproduction are chosen using roulette-wheel selection on the *match subsets* (see below) using accuracy as the reproductive fitness criterion.

Classifier Execution Cycle

On each classifier execution cycle, an input vector is read in from the environment. If this vector is not matched, a cover operator is then applied. However, not all matched classifiers are fired since it is likely that the match set will contain competing as well as cooperating (in the sense that they provide an accurate aggregate output) fuzzy classifiers. Instead, the match set is divided into match subsets, each comprising classifiers whose output membership functions are adjacent (of course the match subset will sometimes contain only a single classifier). The match ssubet fired is chosen using roulette wheel selection based on the aggregate strengths of the classifiers in the match subset. Fuzzy inference and then defuzzification are then applied to determine the classifier system output. Any environmental reward obtained is then used to update the fired classifiers' strengths, predictions and accuracies.

Conclusions and Further Work

A framework for the development of a novel fuzzy classifier system which employs a tree-based classifier representation together with GP as the rule discovery mechanism has been outlined. The fuzzy classifier system described uses accuracy based fitness in an attempt to co-evolve coordinated and general (but not overly general) classifiers. Clearly this framework is at an early stage and requires further investigation. A number of additional features may be incorporated including :

- the use of internal memory (e.g. some form of message list) to deal with environments when current environment state depends upon past actions/states as well as the current one;
- the use of learning methods such as fuzzy Q learning to deal with environments when action rewards are delayed;
- the automatic learning of fuzzy set membership functions.

References

Ahluwalia M. and Bull L. 1999. A Genetic Programming based Classifier System. *Proceedings of the Genetic and Evolutionary Computation Conference(GECCO)*, 11-18, San Francisco CA: Morgan Kaufmann.

Akbarzadeh M.-R., Kumbla K., Tunstel E. and Jamshidi M. 2000. Soft Computing for Autonomous Robotic Systems. *Computers and Electrical Engineering* (26), 5-32.

Bastion A. 2000. Identifying Fuzzy Models Utilizing Genetic Programming. *Fuzzy Sets and Systems* (113), 333-350.

Bentley P.J. 2000. "Evolutionary my dear Watson" - Investigating Committee-based Evolution of Fuzzy Rules for the Detection of Suspicious Insurance Claims. *Proceedings*

of the Genetic and Evolutionary Computation Conference (GECCO), 702-709.San Francisco, CA: Morgan Kaufman.

Bonarini A. 1997. Anytime Learning and Adaptation of Hierarchical Fuzzy Logic Behaviours. *Adaptive Behaviour* 5(3-4):281-315.

Bonarini A. 2000. An Introduction to Learning Fuzzy Classifier Systems. In P.L. Lanzi, W. Stolzmann and S.W. Wilson (Eds.), *Learning Classifier Systems- from Foundations to Applications*, Lecture Notes in Artificial Intelligence, 83-104. Springer Verlag Berlin Heidelberg, Germany.

Carse B., Fogarty T.C. and Munro A. 1996. Evolving Fuzzy Rule-based Controllers using Genetic Algorithms. *Fuzzy Sets and Systems* 80(3):273-293.

Carse B., Fogarty T.C., Munro A. 1998. Artificial Evolution of Fuzzy Rule Bases which Represent Time: a Temporal Fuzzy Classifier System. *International Journal of Intelligent Systems, 13(10/11)*:905-927.

Edmonds A.N., Burkhardt D. and Adjei O. 1995. Genetic Programming of Fuzzy Logic Production Rules. *Proceedings of the 2nd IEEE International Conference on Evolutionary Computation*, 765-770. IEEE Piscataway, NJ.

Glorennec P.Y. 1996. Constrained Optimisation of FIS using an Evolutionary Method. In (Herrera and Verdegay, 1996).

Herrera F. and Verdegay J.L. (Eds.) 1996. Genetic Algorithms and Soft Computing (Studies in Fuzziness, 8). Heidelberg Germany: Physica Verlag (Springer Verlag).

Holland J.H. 1988. Escaping Brittleness: The Possibilities of General Purpose Machine Learning Algorithms applied to Parallel Rule-based systems. In: Michalski R.S., Carbonell J.G and Mitchell T.M. (Eds.), *Machine Learning: an Artificial Intelligence Approach*, vol.2. Kaufmann, Los Altos, California, 1988.

Hwang W. and Thompson W. 1994. Design of Fuzzy Logic Controllers using Genetic Algorithms. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems*, 1383-1388, Piscataway, NJ: IEEE Computer Press

Karr C. L. 1991. Applying Genetics to Fuzzy Logic. *AI Expert* 6(3):38-43.

Ke J.Y., Tang K.S. and Man K.F. 1997. Genetic Fuzzy Classifier for Benchmark Cancer Diagnosis. In *Proceedings of the 23rd International Conference on Industrial Electronics, Control and Instrumentation (IECON97)*, 1063-1067.

Kinzel J., Klawonn F. and Kruse R. 1994. Modifications of Genetic Algorithms for designing and optimising fuzzy controllers. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, 28-33, Piscataway, NJ: IEEE Computer Press.

Koza J.R. 1992. *Genetic Programming*, MIT Press.

Lanzi P.L. and Perrucci A. 1999. Extending the representation of classifier conditions: from messy coding to S-expressions, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 345-352. San Francisco CA:Morgan Kaufmann.

Lee M.A. and Takagi H. 1993. Integrating Design Stages of Fuzzy Systems using Genetic Algorithms. *In Proceedings of the IEEE International Conference on Fuzzy Systems*, 612-617, Piscataway, NJ:IEEE Computer Press.

Liska J. and Melsheimer S. 1994. Complete Design of Fuzzy Logic Systems using Genetic Algorithms. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems*, 1377-1382, Piscataway, NJ: IEEE Computer Press.

Mamdani E.H. 1974. Applications of Fuzzy Algorithms for Control of a Simple Dynamic Plant. *Proceedings of the IEE*, 121(12):1585-1588.

Parodi A. and Bonelli P. 1993. In *Proceedings of the Fifth International Conference on Genetic Algorithms* 223-230. San Mateo, CA:Morgan Kaufman.

Rahmoun A. and Benmohamed M. 1998. Genetic Algorithm Methodology to Generate Optimal Fuzzy Systems. *IEE Proceedings on Control Theory Applications* 145(6):583-587.

Tang K., Man K., Liu Z. and Kwong S. 1998. Minimal Fuzzy Memberships and Rules using Hierarchical Genetic Algorithm. IEEE Transactions on Industrial Electronics 45(1):162-169.

Valenzuela-Rendon M. 1991. The Fuzzy Classifier System: a Classifier System for Continuously Varying Variables. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, 346-353, San Mateo, CA:Morgan Kaufman.

Wilson S. W. 1995. Classifier Fitness based on Accuracy. *Evolutionary Computing* 3(2):149-175.