An Evolutionary Computing Approach to Building Intelligent Frauds Detection Systems

JungWon Kim, Peter Bentley (University College of London)
Jong Uk Choi(Sangmyung University)

ABSTRACT

Frauds detection is a difficult problem, requiring huge computer resources and complicated search activities. Researchers have struggled with the problem. Even though a few research approaches have claimed that their solution is much better than others, research community has not found 'the best solution' well fitting every fraud. Because of the evolving nature of the frauds, a novel and self-adapting method should be devised.

In this research a new approach is suggested to solving frauds in insurance claims and credit card transaction. Based on evolutionary computing approach, the method is itself self-adjusting and evolving enough to generate a new set of decision-making rules. We believe that this new approach will provide a promising alternative to conventional ones, in terms of computation performance and classification accuracy.

1. An Intelligent Financial Fraud Detection System

This research aims to discover fraudulent patterns in a large insurance and credit card transaction databases. This aim is achieved by developing a hybrid Fuzzy-Genetic programming system. The developed system is able to classify home insurance claims and credit card transactions into "suspicious" and "non-suspicious" classes. This section describes the details of developed system and its evaluation results based on various evaluation criteria.

John Koza (1992) developed a genetic programming(GP) for the purpose of automatic programming, which allows computer programs to evolve by themselves. GP differs from other evolutionary algorithms in three main aspects: individuals are represented by tree-structure, crossover normally generates offspring by concatenating random subtrees from the parents, and individuals are evaluated by executing them and assessing their function. Like all evolutionary algorithms, GP maintains populations of solutions. These are evaluated, the best are selected and 'offspring' that inherit features from their 'parents' are created using crossover and mutation operators. The new solutions are then evaluated, the best are selected, and so on, until a good solution has evolved, or a specific number of generation have passed.

There are some recent works which employe especially GP for the evolution of classification rules (Koza et al, 1998), (Raymer et al., 1996), (Ryan et el., 1998), and Ryu and Eick, 1996). It takes advantage of the flexibility of GP that can adopt various functions sets, such as negation, larger-than, etc for a rule operator set. This flexibility allows its phenotypes, classification rules, to express more complex conditions and results in producing a smaller number of rules. However, this expressive power often generates rules which are very difficult to understand and thus the intelligibility of evolved rules is very low. This research focused on tackling this problem. Our system chooses the GP as its main rule learning engine mainly because of its expressive power. We provides the intelligibility of evolved rules by controling crossover points and mates for applying genetic operators. The more details about these methods will be discussed in Section 2, The Evolutionary-Fuzzy Evolver for financial fraud detection.

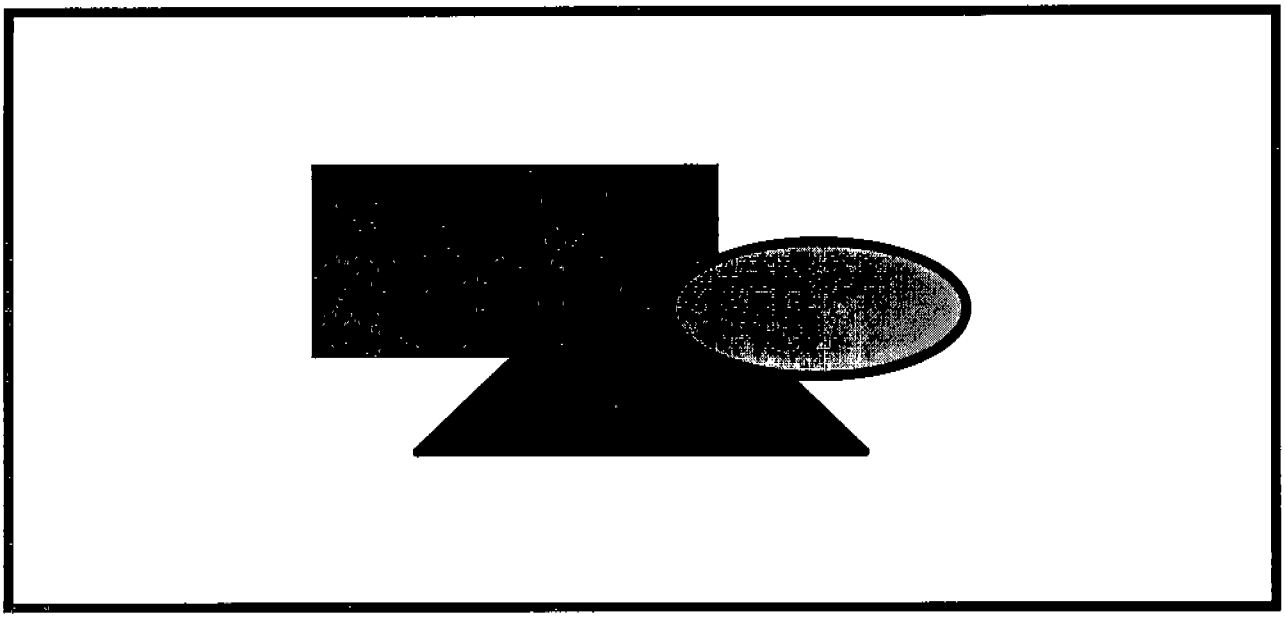
2. The Evolutionary-Fuzzy Evolver for Financial Fraud Detection

The system developed in this research comprises two main components: a Genetic Programming(GP) search engine and a fuzzy expert system. The overall system overview is illustrated in figure 1. This chapter describes the system overview of developed fuzzy rule evolver and the details about this system.

2.1 The System Overview

Before the details of evolutionary-fuzzy evolver system are described, the overview of this system is briefly described. The system starts by taking training data, which has both "fraudulent" and "non-fraudulent" class examples. The training data is immediately passed to the clusterer and the clusterer classifies each attribute(=column) values into three groups. The system

describes each attribute value with one of these fuzzy sets: 'LOW', 'MEDIUM' and 'HIGH' and the clustering stage is necessary to define the possible attribute value ranges of these three fuzzy sets. The cluster passed three generated clusters of each attribute to the fuzzy expert system. The fuzzy expert system defines fuzzy membership functions and their possible ranges based on the generated clusters.. The fuzzy expert system fuzzifies the attribute values of each training item according to the fuzzy membership functions. Then, a GP engine starts evolution by being seeded with random genotypes. The generated genotypes are mapped into phenotypes, which are fuzzy rules. These fuzzy rules are evaluated by being applied to the fuzzified training data in the fuzzy expert system. The results of this procedure return to the GP engine as the defuziffied scores and these scores are evaluated by fitness functions. The estimates fitness scores allow the GP to select fitter phenotypes and reproduce their genotype offsprings. These new genotypes are passed the fuzzy expert system and repeates their evaluation, selection and reproduction until the evolved rules show the certain performance or maximum number of repeats(=generations).



When started, the system first clusters each column of the training data into three groups using a one-dimensional clustering algorithm. A number of clusterers are implemented in the system, including C-Link, S-Link, K-means (Hartigan, 1975) and a simple numerical method (in which the data is sorted, then simply divided into three groups with the same number of items in each group). This paper investigates the last two of these methods in the system. Once selected by the user, the same clusterer is used for all learning and testing of the data.

After every column of the data has been successfully clustered into three, the minimum and maximum values in each cluster are found, see fig. 2. These values are then used to define the domains of the membership functions of the fuzzy expert system.

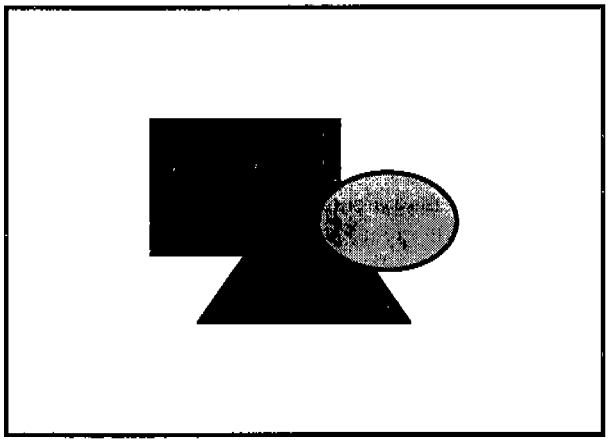


Figure 2: Data is clustered column by column to find the fuzzy membership function ranges.

2.2 Define Fuzzy Membership Functions

The fuzzy rules are used in this research for increasing

the intelligibility and we select the simplest and the most comprehensive three fuzzy sets.: 'LOW', 'MEDIUM' and 'HIGH'. In order to use these linguistic terms directly to collected numeric data, the clustering stage described in the previous section was necessary. The data attributes have wide ranges of values and they are clustered into three groups using a k-mean clustering algorithm. Since each attribute has different range of values and features, the ranges of generated clusters for an individual attribute are very different The three cluster ranges of single attribute are used for defining the 'degree of membership' of three fuzzy sets for the attribute. This results in providing various ranges of fuzzy set definition (more precisely fuzzy membership function values) according to a given attribute.

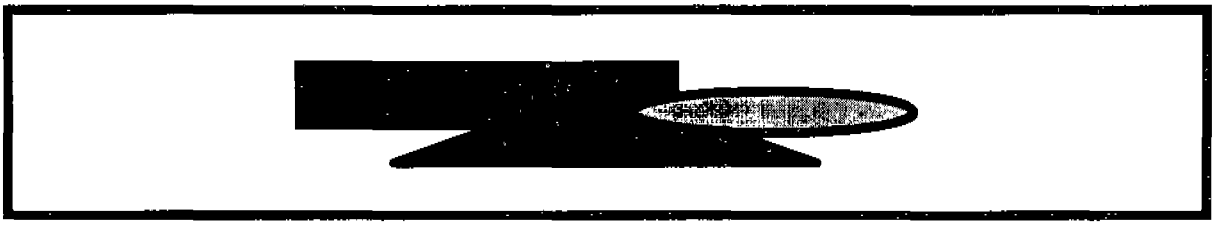


Figure 3: The three types of membership functions used by the system: non-overlapping (left), overlapping (middle), smooth (right).

As we introduced before, the degree of membership for each fuzzy set is defined by a given fuzzy membership function and there are several different shapes of fuzzy membership functions. The system developed in this research provide three various fuzzy membership functions: 'non-overlapping', 'overlapping' and 'smooth', shown in figure 3. The first two functions are standard trapezodal functions and the third one returns a smoother, more gradual set of 'degree of membership' by taking the arctangent of the input.

The different function shapes of these three different functions determines the various definition of each fuzzy set. For the 'non-overlapping' functions, when a specific attribute value belongs to a cluster, they return the membership value 1.0 to the corresponding fuzzy set and 0.0 to the other two fuzzy sets. For instance, if an attribute value is included in the cluster having the lowest range of values, this value would be fuzzified into (1.0, 0.0, .0.0) for 'LOW', 'MEDIUM' and 'HIGH' fuzzy sets, respectively. Because the shape of 'non-overlapping' fuzzy membership function restricts the output areas between two adjacent fuzzy sets to be nearly non-overlapped, it is very usual for given attribute value falling into strictly only one fuzzy set. In other words, it generates the identical fuzzy membership values (1.0, 0.0, .0.0), (0.0, 1.0, 0.0) or (0.0, 0.0, 1.0) to most of attribute values. In contrast, the second fuzzy membership 'overlapping' function widen the overlapping areas between neighboring fuzzy sets and more versatile fuzzified membership values can be expected. This means that a value towards the outer degree of the 'LOW' fuzzy set might be fuzzified into (0.8, 0.2, 0.0) instead of (1.0, 0.0, 0.0). Finally, the last 'smooth' functions expands the overlapping areas among three fuzzy sets even more. For instance, even when a value resides on the center of 'LOW' fuzzy set area, this fuzzy membership function returns the fuzzified value (0.98, 0.02, 0.0) signaling somehow the value locating at a much nearer point from the 'MEDIUM' set than the 'HIGH' set.

In summary, when a training example is provided to the system, it fuzzifies all attributes of this example according to a selected fuzzy membership function. and the fuzzy set ranges are determined by clusterers generated using the k-mean clustering algorithm.

2.3 Evolving Rules

When fuzzified data is passed to a GP engine and the fuzzy rule evolver starts the rule evolution, The rule evolution by the GP is required to pass a number of important stages to achieve its task, generating fuzzy rules classifying fraudulent data and non-fraudulent data.

2.3.1 Genotypes and Phenotypes

The operation of natural evolution can be understood by the evolution of a set of coded instructions for how organisms should be grown (Bentley, 97). That is to say, the genetic operators to drive the natural evolution are not applied directly on organisms. Rather, they are applied on a set of coded instructions: DNA. According to these instructions, the organisms evolve and appear to have various types of patterns. In order to understand the natural evolution clearly, it is necessary to distinct DNA from the evolved organisms based on their DNA. The former is known as a genotype and the latter is called as a phenotype.

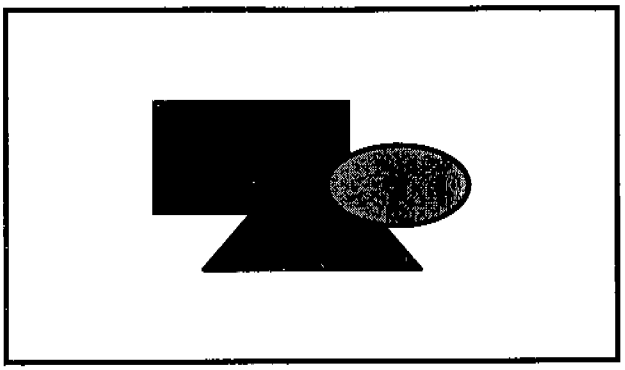


Figure 4: An example genotype used by the system.

The genotypes consist of variable size trees, where each node is comprised of a binary number and a flag indicating whether a node is binary, unary or a leaf. A binary node has two branches (left and right), a unary node has one branch (left or right) and a leaf node is a terminating node of given branch. When evolution starts, genotype genes with tree structures are randomly generated and the created genotype genes are usually have no more than three binary and four unary nodes. We restrict of tree size at the start to avoid the generation of unnecessarily long depth of trees. These genotype genes are immediately mapped into phenotype genes in order to be evaluated. Fig.4 shows the mapping of genotype genes in Fig. 4 into the phenotype genes, which are in a form of fuzzy rules:

(IS_MEDIUM (Height OR IS_LOW Age) AND IS_MEDIUM Age).

system employes two binary functions: 'OR' and 'AND', four unary functions: 'NOT', 'IS_LOW', 'IS_MEDIUM', 'IS_HIGH'. Each leave of tree indicates a single attribute and the system restricts the number of leaves up to 256. An individual genotype tree is easily interpreted into a fuzzy rule by reading a given node type and translate binary value into tenary value.

2.3.2 Rule Evaluation

Every chromosome containing phenotype genes, which is a fuzzy rule, is evaluated by applying it to the fuzzified training data. This work is performed by a fuzzy expert system. The results of this work is returned in a defuzzified score between 0 and 1 for every fuzzified data item. Then, these defuzzified scores are assessed by four different fitness functions:

- Low misclassification rate: minimize the number of misclassified examples. The misclassified examples are those which have the defuzzified scores larger than 0.5 when they are 'non-fraudulent' items. In fact, this is the case when 'non-fraudulent' case is misclassified as 'fraudulent' case. The system regards the item with the defuzzied score close to 1 as 'fraudulent' case.

- Maximize the distinction between 'non-fraudulent' and 'fraudulent' classes: to distinct two classes, this fitness function measures the average defuzzified scores for correctly classified 'fraudulent' cases and the average defuzzified scores for correctly classified 'non-fraudulent' cases. As the difference of these two average scores increase, this fitness score becomes higher.

- Assign a high priority to detect 'fraudulent' items more: the system considers the detection of 'fraudulent' cases more importantly. Thus, this fitness function assess the sum of scores for 'fraudulent' cases and assign higher fitness value when this score gets higher.

- Increase the intelligibility: in order to increase the intelligibility, the system penalizes the length of any fuzzy rules that has more than four identifier, which are binary, unary or leaf nodes. By doing so, the system ensures that the evolved rules always have the readable length of condition parts and also can prevent the bloat caused by the GP.

One distinct feature of our evolutionary fuzzy evolver is that it has multiple fitness functions to collectively satisfy the desired function of the system. Most of real world problems require more than one subtask to be fulfilled due to its complicated nature. However, this feature causes a major problem for the standard GA. It usually cannot cope with more than one fitness value per phenotype(Goldberg, 1989). The standard GA equips only one fitness value for every individual in the current population of solutions. This value represents how well its corresponding solution satisfies the goal of a given problem. According to the estimated fitness values, the GA can select fitter ones and gives higher chances to reproduce their offsprings which inherit some features of original solution. The problem is raised when the GA tries to select the fitter ones based on calculated fitness values. In order to do so, the GA should apprehend the single relative fitness value of each candidate solution for comparison even when each individual has multiple fitness values. The question is how can we make the GA to define a single fitness value that represents the aggregation of multiple fitness values accurately?

Bentley and Wakefield(1997) developed the Sum of Weighted Global Ratios(SWGR) method to tackle this problem and showed its successful results in their evolutionary design system. The system developed in this research also employs this approach. To define the overall fitness values from multiple fitness values, the SWGR first scales each fitness value using the *effective ranges* of each function. Since the input of each fitness

function shows different range of possible values, this scaling is perforemed first. For example, for a given individual *I*, the scaling is simply done by

$$fitnessRatio = \frac{fitnessValue_i - \min(fitnessValue)}{\max(fitnessValue) - \min(fitnessValue)}$$

Then, the normalized fitness values are multiplied by *importance* values defined by system users. The basic notion of importance is that even though the final goal is achieved by satisfying the multiple subtasks, the relative importance of each subtask will be different and this difference can be defined by the users according to their perceived task priority. For instance, if the user specified that the second criteria should be twice as important, all fitness ratios corresponding to the second criteria are simply multiplied by two. Finally, SWGR sums these importance weighted global fitness ratios and generates the single global fitness values.

2.3.3 Rule Generation

Genetic Operators

Child rules are generated using one of two forms of crossover. The first type of crossover emulates the single-point crossover of genetic algorithms by finding two random points in the parent genotypes that resemble each other, and splicing the genotypes at that point. By ensuring that the same type of nodes, in approximately the same places, are crossed over, and that the binary numbers within the nodes are also crossed, an effective exploration of the search space is provided without excessive disruption (Bentley & Wakefield, 1996). The second type of crossover generates child rules by combining two parent rules together using a binary operator (an AND or OR). This more unusual method of generating offspring (applied approximately one time out of every ten instead of the other crossover operator) permits two parents that detect different types of suspicious data to be combined into a single, fitter individual. Mutation is also occasionally applied, to modify randomly the binary numbers in each node by a single bit.

Selection

The GP system employs population overlapping, where the worst *Pn*% of the population are replaced by the new offspring generated from the best *Pm*%. Typically values of *Pn* = 80 and *Pm* = 40 seem to provide good results. The population size was normally 100 individuals.

Modal Evolution

Each evolutionary run of the GP system (usually only 15 generations) results in a short, readable rule which detects some, but not all, of the suspicious data items in the training data set. Such a rule can be considered to define one mode of a multimodal problem. All items that are correctly classified by this rule (recorded in the modal database, see figure 1) are removed and the system automatically restarts, evolving a new rule to classify the remaining items. The parameter *nich size* specifies the number of "fraudulent" data items sought to be classified in each run. This enables monitoring of over-fitting: the fitness is correlated with the number of class members classified by a rule as well as the number miss-classified.

Modal Re-Evolution

In addition to the process of modal evolution, the system re-examines each mode already classified by a rule; it attempts to improve the rule by ignoring all data except that characterized (and mis-classified) by the rule already. This provides a shrinking environment, with the associated gains: a 'purer' gene pool of solutions for each archtype is facilitated, and accelerated search.

Nested Evolutionary Search

After shrinking the environment(reducing the number of claims again which a rule is tested) the system can recluster and carry out a finer search. This process of modal evolution continues until every suspicious data item has been described by a rule. However, any rules that misclassify more items than they correctly classify are removed from the final rule set by the system.

3. Experiments and Results

3.1 Data

As with any real-world problem, classification of real data is often far removed from the clean, perfect world of mathematical theories. Data is usually noisy, inconsistent and sometimes inadequate. Even though intelligent techniques such as GP and fuzzy logic can handle such characteristics better than many approaches, significant data preprocessing will always be required.

3.1.1 Preprocessing the Data
(**Lloyds/TSB** INSURANCE DATA)

The insurance data used for this work was no exception. The data came from numerous sources within the bank, resulting in two somewhat incompatible files. One file contained 98 cases of suspicious insurance claim, each with 73 fields (this was assembled from numerous different files provided). The other contained 20,000 cases of unknown insurance claims (that might or might not be suspicious), each with 36 fields. The fields comprised items such as policy number, claim number, date of birth, policy type, etc. However, the two files had very few fields in common. Even after constructing some new fields by processing others in different formats, only 14 common fields in both files could be found.

Once all non-corresponding fields were removed, we were left with two files, one containing 98 claims, each

with 14 fields, the other containing 20000 claims, each with the same 14 fields. The data for every pair of fields was then converted into the same format (for example, dates were initially stored in different formats, different codes were used, etc). Missing values in the files were replaced by random values within the range of normal values for each field. (Attempting to classify data with missing values is difficult, so it is simpler to fill the gaps with random values. This has the effect of adding a small level of noise to the data in this case 1.07% overall. However, the distribution of missing values, and hence noise per field was not even it varied from 0% to 17%. By keeping a record of the percentage noise per field, the reliability of evolved rules that use the noisiest fields can then be reduced. Note that additional noise in the form of errors within the data was also evident.)

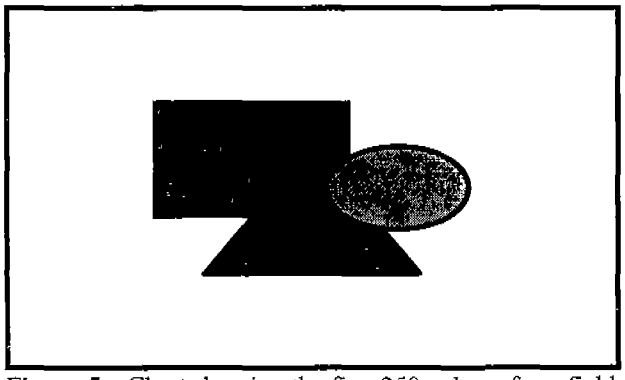


Figure 5 Chart showing the first 250 values, for a field related to the date. Note how the suspicious values in the first 49 are much lower on average than the unknown values in claims 50 upwards.

In an attempt to extract more information from the data, and give the classifier a better chance of success, six new fields were created by processing existing fields. For example a new field called days before claiming was constructed by subtracting values in the field accident date from the values in the field notified date.

A training and test data file was constructed, each containing 49 suspicious claims and 10000 unknown (alternate claims taken from the original files). A series of experiments were then performed using the evolutionary fuzzy system. The results were suspiciously good indeed, accuracy was 100%. From these experiments it became clear that inconsistencies in the data were proving considerably more useful as indicators of fraud than anything else. The disparity was mainly caused by the fact that the 98 cases of suspicious claim were gathered over a period of some years, whilst the unknown data was gathered over a recent period of three months. Any field that varied according to the date was therefore lower, on average, for the suspicious fields compared to the unknown. By plotting charts of each field, it was simple to discover that this adversely effected six of fields, e.g. see fig. 5.

While it is possible that variations on the frequency of fraud may depend on absolute values of dates (e.g. perhaps fraud becomes more likely during a particular month of a year, or following a television programme on how to do fraud), this was seen as unlikely. It was therefore more desirable to attempt to find more generic indicators of fraud, not those dependent on absolute times or specific policy numbers. Consequently, all six fields were deleted (and a seventh which had the same value for all claims was also removed), leaving thirteen fields in each data item. Information was not lost, however. The new fields mentioned earlier contained *relative* date information, so the data contained within five of the deleted fields was still available (with the benefit that the time biases were removed, as differences between fields were used, rather than absolute values).

Credit Card Company DATA

The data used in this work was gathered from a credit card company in England. Even though the company provided real credit card transaction data for this research, it required that the company name was kept confidential. The data was gathered from January to December of 1995 and a total of 4000 transaction records were provided, each with 96 fields. 62 fields were selected for the experiments. The excluded 34 fields were regarded as clearly irrelevant for distinguishing the credit status. (Examples include the client code number and the transaction index number.) The details of selected field names were not allowed to be reported. In order to allow the fuzzy rule evolution of the system, the collected data was labeled as suspicious or non-suspicious. These labels were made by following the heuristic used in the credit card company. Specifically, when the customers payment is not overdue or the number of overdue payment is less than three months, the transaction is considered as non-suspicious, otherwise it is considered suspicious.

To prepare a training set and a test set, we employed a simple cross-validation method. We held one-third of the data for testing and used the remaining two-thirds for training. The system executed its rule-evolution three times on three different training data sets. For each run, the system replaced the training set with the other third of the data set. This cross-validation was performed in order to ensure the evolved rule sets were not biased by a certain group of training set. By comparing the three different evolved rules based on three different groups of training data set, the final rule set is expected to represent the features of the entire data set. Unfortunately, the distribution of collected credit card transaction data was not even for each class. It had a larger number of examples for the "non-suspicious" class than for the "suspicious" class. The total number of items belonging to the smaller size of "suspicious" class was 985. This number is large enough to be divided into three subsets.

Thus, the four committee members with identical experiment setups were run three times on each data subset respectively. The examples included in each set are shown in Table 1.

Table 1. Credit card data distribution for three experiments. The number in this table shows the IDs of examples belonging to each set. Exp stands for the experiment.

Exp	"SUSPICIOUS"		"NON-SUSPICIOUS"	
	Training	Test	Training	Test
1	1-656	657-985	1-2000	2000-3015
2	329-985	1-328	1001-3015	1-1000
3	657-985 & 1-328	329-656	2001-3015 & 1-1000	1001-2000

3.2 Experiments

With the requirements for a good fraud-detection system in mind, this section describes a series of experiments designed to evaluate these key capabilities of the system. The experiments investigate three aspects of the system: the effect of using different membership functions and fuzzy operators, the effect of using different clusterers, and the ability of the system to cope with noisy data. For all three sets of experiments, the intelligibility of results, processing time, and accuracy of detection are assessed.

3.2.1 Experiment Setup

To allow comparison of this system with other techniques reported in the literature, the fuzzy rule evolver was applied to two standard data sets for all experiments: the Iris and Wisconsin Breast Cancer data sets.

The Iris data is perhaps the best known database to be found in the pattern recognition literature according to the information provided by UCI with the data and it comprises a simple domain of 150 instances in three classes, each of 50 items. Data items have four attributes; there are no missing values. Because the 'Setosa class' is linearly separable from the other two classes, for all experiments the system was set the harder task of detecting the 'Virginica' class from the 'Versicolour' and 'Setosa' classes combined. Training and test data files were prepared by splitting the data set into two (taking alternate data items for each file). Misclassification rates for this data set are normally reported as 0% for the Setosa class and very low for the other classes in the literature e.g. (Dasarathy, 1980).

The Wisconsin Breast Cancer data is a more complex data set, comprising 699 instances in two classes: Malignant (241 data items) and Benign (458 items). There are 16 missing values in the data, which were filled by random numbers. The training and test data sets were constructed by splitting the file into two, taking alternate values. (For the sake of symmetry, one Malignant item was discarded and two Benign items duplicated, resulting in two sets of 350 data items, each with 120 Malignant.) Results reported in the literature include accuracies of 93.5%, 95.9% (Wolberg, and Mangasarian, 1990), and 92.2% (Zhang, 1992).

50 trials were run for each experiment, with the average and best accuracies reported here. Percentage accuracy of detection was found by calculating:

$$100 - \frac{100(MisclassifiedItems + UnclassifiedItems)}{TotalItems}$$

was measured in terms of the average number of rules evolved - the fewer the rules, the more intelligible the result. Average processing speed was measured in seconds (and includes the negligible time taken to apply the completed rule set to both data sets).

The fitness functions reported in section 3.3.2 were used without change for all experiments. Importance rankings (Bentley & Wakefield, 1997) were set as 0.5, 2.0, 1.0 and 0.5 for fitness functions one to four, respectively. Mutation of a single bit occurred with a probability of 0.001 in each genotype. Population sizes of 100 were used, and each modal evolutionary run was for exactly 15 generations. The K-Means clusterer was used in the system (unless otherwise stated). Experiments were run on a PC with a 233Mhz AMD K6 processor.

3.2.2 Experiment 1: Investigating The Effects of Committee-Decisions for Insurance Data

As should be apparent, the task of detecting genuine patterns of fraud using the data provided was not trivial. Indeed, although the data was now in a fit state to be used by a classifier, there still remained the problem of the unknown data set. Lloyds TSB suggested that up to 5% of the items in this set might be suspicious, but which claims and exactly how many was unknown. To tackle this problem, three sets of experiments were performed with the committee decision system. The first experiment assessed the ability of the system to find rules indicative of suspicious items, without those patterns describing any unknown items. The second experiment assessed how well the system could find suspicious rules that also detected up to 5% of the

unknown items. The third experiment assessed the ability of the system to find rules that detected suspicious items and up to 10% of the unknown items. (Note that although the system does report which claims in the unknown data set were found to be suspicious, these results cannot be provided here.)

Each experiment used four setups of the system:

1. standard fuzzy logic with non-overlapping membership functions
2. standard fuzzy logic with overlapping membership functions
3. membership-preserving fuzzy logic with overlapping membership functions
4. membership-preserving fuzzy logic with smooth membership functions

All four committee members were trained on one file and tested on the other, then trained on the second and tested on the first. This resulted in 24 different rule sets being generated for this problem, each with different levels of intelligibility and accuracy.

Table 2 Intelligibility (number of rules) and accuracy (number of correct classifications of suspicious items) of rule sets for test and training data. Accuracy rates are listed as n, m where n = number out of 49 correctly classified in class 1, m = number classified out of 10,000 in class 2. Results are given for training on file1, testing on file2 and training on file2, testing on file1.

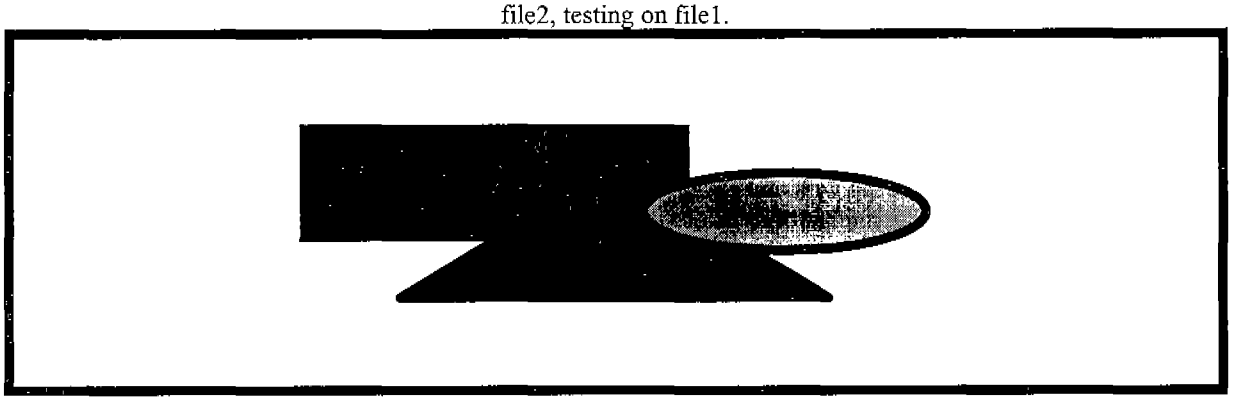


Table 3 Best results as reported by committee decision maker.

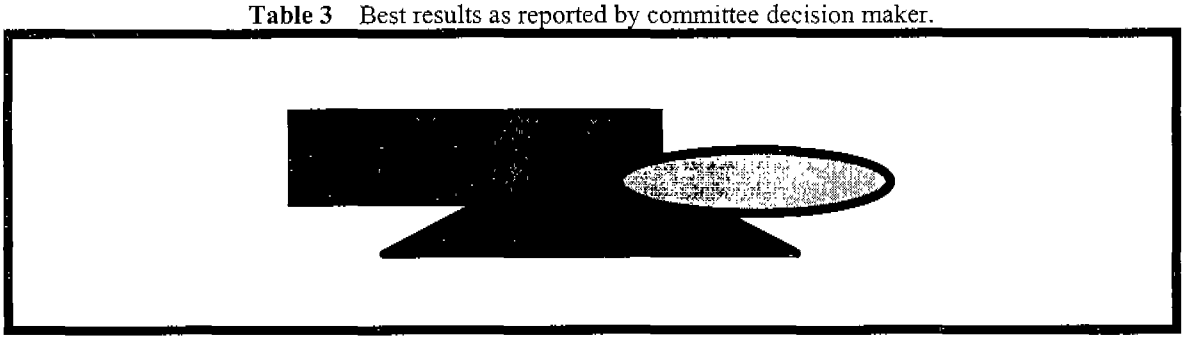


Table 4 Frequencies of fields in all rule sets and reliability of fields (based on noise caused by filling missing values). Note that NOT IS_X field is expanded to IS_Y Field or IS_Z Field and IS_LOW IS_LOW is translated to IS_HIGH for mp-fuzzy logic.

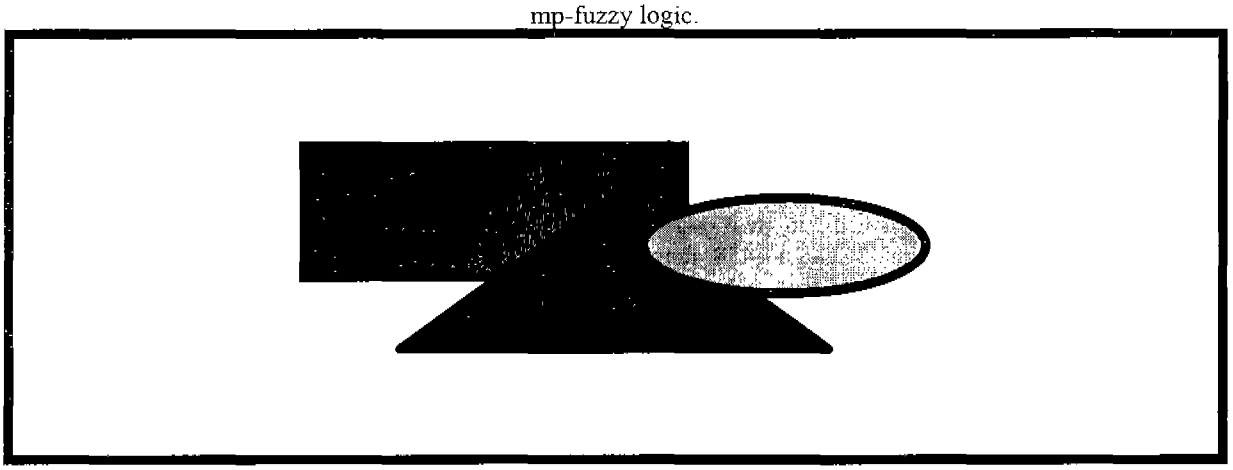


Table 2 and 3 present the results of the experiments. It should be apparent in Table 1 that no committee member managed to find useful rules that detect 0% suspicious claims in the unknown set indeed most failed to generate any valid rules at all. When up to 5% or 10% suspicious claims are assumed to exist in the unknown data set, accuracy rates increase dramatically. As Table 2 explains, committee members [A] and [D] provide the most accurate and intelligible classifications for all experiments with this data. The best accuracy overall is achieved by [A], finding 61 out of 98, or 62% of the suspicious claims, whilst suggesting that 1339 out of 20000, or 6.7% of the unknown claims are also suspicious. But the most accurate and intelligible rule sets are generated by [D], with most rule sets containing just a single rule. Overall, the best rule set as reported by the committee decision maker is:

(IS_LOW Field8 OR Field3)

which can be translated as: If either the value for field8 is low or the value for field3 is high, then in 57% of observed cases the claim will be suspicious. This rule suggests that 3.8% of the unknown claims are suspicious.

Further analysis can be performed by examining the occurrences of fields in the evolved rules, see table 4. In general, the tally of field occurrences in the rules as shown above indicates that suspicious claims seem to be more likely when:

Fields 1, 5, 7, 9 and 13 are medium or high
Fields 2,3,4 and 6 are high
Field 8 is low or high
Fields 11 and 12 are low or medium

Interestingly, the only field with significant levels of noise Field10 is hardly used for classification in the rules. The table also shows that Field3 seems to provide the single best indication of suspiciousness. Indeed, even used on its own, the rule:

IS_LOW IS_LOW Field3
which in mp-fuzy logic should be translated as:
ISVERYHIGH Field3

is capable of detecting 54 out of 98 suspicious claims.

3.2.3 Experiment 2: Investigating The Effects of Committee-Decisions for Credit Card Data

Three sets of experiments were performed with the committee decision system and the four different setups of fuzzy rule evolver were run for each experiment:

1. Standard fuzzy logic with non-overlapping membership functions
2. Standard fuzzy logic with overlapping membership functions
3. Membership-preserving fuzzy logic with overlapping membership functions
4. Membership-preserving fuzzy logic with smooth membership functions

All four committee members were trained on one selected training set and test set. This resulted in different rule sets being generated for this problem, each with different levels of intelligibility and accuracy.

Table 5 presents the results of the experiments. The accuracy of the system is described by a True Positive (TP) prediction rate and a False Negative (FN) error rate. The TP is the rate that the predicted output is "suspicious" class when the desired output is "suspicious" class. The FN is the probability of which the predicted output is "suspicious" when the desired output is "non-suspicious" class. The desired system will have a high TP and a low FN.

As Table-5 explains, committee member [B] provides the most accurate and intelligible classifications for all experiments with this data. The best accuracy overall is achieved by [B], detecting 100% of the suspicious claims for both on the training and the test set, whilst showing that 5.79% of false negative error, which is relatively low. In addition, the most accurate and intelligible rule sets that are generated by [B] contain just three rules. Overall, the best rule set as reported by the committee decision maker is for experiment 2:

(IS_LOW field57 OR field50)
IS_MEDIUM field56
(field56 OR field56)

and for the experiment 3:

(Filed49 OR Field56)
(IS_LOW Field26 OR field15)
IS_MEDIUM field56

These best rule sets are clearly dominated by the field56. This implies that this field seems to be the single best indicator of suspicious case. In summary, the prediction results of these best rule sets are satisfying in terms of the accuracy and intelligibility.

Table 5 Intelligibility (number of rules) and accuracy (number of correct classifications of suspicious items) of rule sets for test and training data. **R** shows the number of rules in the generated rule set and **TP** and **FN** is represented in %.

Exp	[A] Fuzzy Logic with non-overlapping MFs					[B] Fuzzy Logic with overlapping MFs					[C] MP-Fuzzy Logic with overlapping MFs					[D] MP-Fuzzy Logic with smooth MFs				
	R	Training		Test		R	Training		Test		R	Training		Test		R	Training		Test	
		TP %	FN %	TP %	FN %		TP %	FN %	TP %	FN %		TP %	FN %	TP %	FN %		TP %	FN %	TP %	FN %
1	3	6.09	3.81	10.4	3.35	2	100	0	100	83.1	16	10.9	5.79	100	100	5	48.6	5.79	42.5	10.3
2	2	44.1	5.79	47.8	9.45	3	100	1.67	99.7	6.38	3	1.37	5.64	99.7	100	10	41.6	5.79	47.6	12.5
3	3	46.8	5.18	46.9	6.09	3	100	5.78	100	5.79	4	1.67	5.64	86.9	100	16	42.7	5.94	42.9	6.40

Another interesting observation is that the results of experiments rapidly change depending on the specific experiment setup. While [B] setup always generated the good rule sets, [C] setup provided almost meaningless rule sets, which showed nearly random prediction results. The setup [D] showed the consistent results, which the differences of TP and FN for both the training and the test sets are within 6%, but the best result is not satisfying. These results show again the large variance of committee member performance and illustrate the validity of the committee-decision maker approach for this problem.

In addition, from [A] and [B]s results, it could be implied that the data set used in the experiment 1 seems to have somewhat different characters from other two data sets. The quite large difference, about 40% for TP in [A] and 80% for FN in [B] represent that the importance of data sampling during the fuzzy rule evolution stage.

4. Conclusion

This research has investigated the use of a genetic programming system to evolve fuzzy rules for the purpose of detecting suspicious data amongst normal data. The system contains many novel elements, including a crossover operator designed to minimize disruption, binary genotype, and a new method for interpreting fuzzy rules designed to preserve all fuzzy set membership values. Consultation with our collaborating company, Lloyds/TSB resulted in a set of evaluation criteria for the system: intelligibility, speed, handling noise and accuracy.

With these aspects in mind, three sets of experiments were performed on the system, using two standard data sets to permit comparison with the literature. The first test investigated the effect of membership functions on the system. By increasing the overlap between fuzzy membership functions and by preserving the information held in the membership values, the results showed that the number of rules needed to classify data could be reduced. This reduction often led to a decrease in accuracy of classification, but this was offset by the dramatically increased intelligibility of output, faster processing time, and better feature selection. The second test investigated the effects of using different clusterers in the system. It was found that a basic clusterer slightly reduced the accuracy of the system, compared to the more complex K-Means approach. The choice of clusterer did not seem to have any consistent effect on intelligibility of output or processing speed. The final test investigated the ability of the system to cope with increasing levels of noise in the data. As one would expect, accuracy of classification was detrimentally affected as noise increased. Interestingly, the intelligibility and processing speed showed no clear trend for increasing levels of noise.

Together, these experiments show:

many factors affect accuracy of classification

intelligibility and processing speed only seem to be affected by the type and use of membership functions - noise and the choice of clusterer seems unimportant.

noisy data causes at best a linear drop in accuracy, and at worst, a fall proportionate to the square of input noise.

As the second stage, this research has described the use of genetic programming to evolve fuzzy rules within a parallel committee decision system. Attention was paid to data preprocessing, describing some of the typical problems associated with real-world data in order to show just how hard this kind of classification becomes. Nevertheless, despite having only 49 suspicious items in the first class to train the system, and an unknown number of suspicious items in the 10000-item second class, performance of the system was good. Given the quality and quantity of the data, accuracy rates of over 60% must surely be regarded as impressive. Indeed it seems very likely that better accuracy would only result in overfitting the meagre training data. In addition, intelligibility rates were excellent with many rule sets comprising a single, understandable rule.

This work shows the benefit of committee decision making. Each of the four different committee members (different setups of the evolutionary fuzzy system) provided different rates of accuracy and intelligibility. The committee decision maker was able to analyse all results and pick the best rule set.

The evolved rules and the table of field frequencies in rules have provided important and interesting

information about the nature of fraud in home insurance claims. Sadly the names of the fields and the true meanings of the rules cannot be reported in this article, but Lloyds TSB have stated that the results were sensible as confirmed by previous analysis, and support the potential for even more useful results with improved data.

Finally, a committee-decision-making evolutionary fuzzy system developed in this work was applied for domestic credit card transaction data evaluation. The results for this real-world problem confirm previous results obtained for real home insurance data. They illustrate that the use of evolution with fuzzy logic can enable both accurate and intelligible classification of difficult data. The results also show the importance of committee-decision making to help ensure that good results will always be generated.

5. Reference

(Bentley, 1997) Bentley, P. J., *Generic Evolutionary Desion of Solid Objects using a Genetic Algorithm*, PhD Thesis, Division of Computing and Control Systems, The University of Huddersfield, 1997.

(Bentley and Wakefield, 1997) Bentley, P. J., and Wakefield, J. P, Finding acceptable solutions in the Pareto-Optimal Ranges using multiobjective genetic algorithms., Chawdhry, P. K., Roy, R., and Pant, R.K., (eds) Soft Computing in Engineering Design and Manufacturing, Springer Verlag London Limited, Part 5, pp.231-240.

(Bezdek and Pal, 1992) *Fuzzy Models for Pattern Recognition,* IEEE Press, New York, 1992.

(Booker et al., 1989) Booker, L. B., Goldberg, D. E., and Holland, J. H., "Classifier Systems and Genetic Algorithms", *Artificial Intelligence*, Vol.40, No.1-3, pp.235-282, 1989.

(Bunn, 1989) Bunn, D., "Forecasting with more than one model." *Journal of Forecasting* v8, pp. 161-166, 1989.

(Chiu, 1997) Chiu, S., "Extracting Fuzzy Rules from Data for Function Approximation and Pattern Classification", in *Fuzzy Information Engineering*, (Ed) Dubois, Prade, Yanger Wiley, 1997.

(Chung and Lee, 1985) Chung, F. L., and Lee, T., "A Fuzzy k-nearest Neighbour Algorithm", *IEEE Transactions on System, Man, and Cybernetics*, Vol. 15, pp.580-585.

(Dasarathy, 1980) Dasarathy, B.V., "Nosing Around the Neighborhood: A New System Structure and Classification Rule for Recognition in Partially Exposed Environments." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-2, No. 1, 67-71, 1980.

(Deb and Goldberg, 1989) Deb, K. and Goldberg, D. E., "An investigation of niches and species formation in genetic function optimization", *Proceeding of the 3rd International Conference on Genetic Algorithms*, pp.42-50, Fairfax, VA, Morgan Kaufmann, 1989.

(De Jong et al., 1993) De Jong, K. A., Spears, W. M., and Gordon, D. F., "Using Genetic Algorithms for Concept Learning", *Machine Learning*, Vol.13, No.2/3, pp.161-188, 1993

(Eberhart et al. 1996) Computational Intelligence PC Tools, Academic Press, London, 1996.

(Flockhar, 1995) Flockhar, I. W., "GA-MINER: Parallel DataMining with Hierarchical Genetic Algorithms" Final Report, EPCC-AIKMS-GA-MINER-REPORT 1.0, Edinburgh Parallel Computing Center, 1995.

(Giarratano and Riley, 1994) Giarratano, J. and Riley, G., *Expert Systems: Principles and Programming*, PWS Publishing Company, Boston, 1994.

(Giodana and Neri, 1996) Giodana, A. and Neri, F., "Search-intensive concept induction", *Evolutionary Computaion,* Vol.3, No.4, pp.375-416, 1996.

(Goldberg, 1989) Goldberg, D. E., *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, 1989.

(Hartigan, 1975) Hartigan, J. A (1975). *Clustering algorithms*. Wiley, NY.

(Hekanaho, 1997) Hekanaho, J., DOGMA: A GA-Based Relational Leaner, TUCS Technical Report No.168, Turku Centre for Computer Science, May, 1997.

(Ishibuchi et al., 1998) Ishibuchi, H., Murata, T., and Nakashima, T., Genetic Algorithm-Based Approaches to Classification Problems, Fuzzy Evolutionary Computation, Witold Pedrycz(Ed), pp.127-153, 1997.

(Janilkow, 1993) Janikow, C. Z., A Knowledge-Intensive Genetic Algorithm for Supervised Learning, *Machine Learning*, Vol.12, pp.189-228, 1993.

(Ko et al., 1994) Ko, C., Fink, G., Levitt, K., "Automated Detection of Vulnerabilities in Privileged Programs by Execution Monitoring". *Proceeding of the 10th Annual Computer Security Applications Conference*, Orlando, FL, pp.134-144, 5-9 Dec. 1994.

(Koza, 1992) Koza, J., *Genetic Programming: On the programming of computers by means of natural selection.*, MIT press, 1992.

(Langley, 1996) Langley, P., *Elements of Machine Learning*, Morgan Kaufmann Publishers, San Francisco, 1996.

(Mallinson and Bentley, 1999) Mallinson, H. and Bentley, P.J. (1999). Evolving Fuzzy Rules for Pattern Classification. In *International Conference on Computational Intelligence for Modelling, Control and Automation - CIMCA'99.*

(Marmelstein and Lamont, 1998) Marmelstein, R. E., and Lamont, G. B., "Evolving Compact Decision Rule Sets", in Koza, J. (Ed.), Late Breaking Papers at the Genetic Programming 1997 Conference, University of Wisconsin, July 22-25, pp.144-150, 1998.

(Mitchell, 1997) Mitchell, T., *Maching Learning*, McGraw-Hill Inc., 1997.

(Pedrycz, 1997) Pedrycz, W. (Ed.), *Fuzzy Evolutionary Computation*, Kluwer Academic Publishers, MA., 1997.

(Potter, 1997) Potter, M. A., *The design and analysis of a computational model of cooperative co-evolution*, PhD Thesis, George Mason University, Fairfax, VI., 1997.

(Raymer et al., 1996) Raymer, M. L., Punch, W. F., Goodman, E.D., and Kuhn, L. A., "Genetic Programming for Improved Data Mining Application to the Biochemistry of Protein Interactions", *Proceeding of the First Annual Genetic Programming Conference*, pp.375-380, Stanford University, 1996.

(Ross, 1995) *Fuzzy Logic with Engineering Applications*, McGraw-Hill, 1995.

(Ryan et el., 1998) Ryan, M. D., and Rayward-Smith, V. J., "The Evolution of Decision Trees", *Proceeding of the Third Annual Genetic Programming Conference*, pp.350-358, University of Wisconsin, Madison, Wisconsin, 1998.

(Ryu and Eick, 1996) Ryu, T., and Eick, C. F.,. "Deriving queries from results using genetic programming",. *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 303--306, Portland, OR, 2.-4. August 1996. AAAI Press, Menlo Park, CA, 1996.

(Smith, 1983) Smith, S. F., "Flexible learning of problem solving heuristics through adaptive search", *Proceeding of the 8th Internaional Joint Conferenece on Artificial Intelligence*, pp.422-425, Karlsruhe, Germany, 1983.

(Wolberg and Mangasarian, 1990) Wolberg, W. H., and Mangasarian, O. L. "Multisurface method of pattern separation for medical diagnosis applied to breast cytology", In *Proceedings of the National Academy of Sciences*, 87, pp.9193--9196, 1990.

(Yu and Bentley, 1998) Yu, T. and Bentley, P., "Methods to Evolve Legal Phenotypes.", In *Proceedings of the Fifth Int. Conf. on Parallel Problem Solving From Nature*. Amsterdam, Sept 27-30, 1998, pp. 280-282, 1998.

(Zhang, 1992) Zhang, J., "Selecting typical instances in instance-based learning". In *Proceedings of the Ninth International Machine Learning Conference*, pp. 470--479. Aberdeen, Scotland: Morgan Kaufmann, 1992.