Genetic Fuzzy Discretization with Adaptive Intervals for Classification Problems

Yoon-Seok Choi
School of Computer Science & Engineering,
Seoul National University
Shillim-dong, Gwanak-gu,
Seoul, 151-742, Korea
yschoi@soar.snu.ac.kr

Byung-Ro Moon
School of Computer Science & Engineering,
Seoul National University
Shillim-dong, Gwanak-gu,
Seoul, 151-742, Korea
moon@soar.snu.ac.kr

Sang Yong Seo
Senior Researcher,
Multimedia Technology Development Team,
KT Marketing & Technology Laboratory
seosy@kt.co.kr

ABSTRACT

We propose a genetic fuzzy discretization method for continuous numerical attributes. Traditional discretization methods categorize the continuous attributes into a number of bins. Because they are made on crisp discretization, there exists considerable information loss. Fuzzy discretization allows overlapping intervals and reflects linguistic classification. However, the number of intervals, the boundaries of intervals, and the degrees of overlapping are intractable to get optimized. We use a genetic algorithm to optimize these parameters. Experimental results showed considerable improvement on the classification accuracy over a crisp discretization and a typical fuzzy discretization.

Categories and Subject Descriptors

I.5 [**Pattern Classification**]: Models—*Fuzzy set*

General Terms

Experimentation, Performance

Keywords

Genetic Algorithm, Fuzzy Discretization, Classification Problem

1. INTRODUCTION

For knowledge discovery from a raw data set, one first preprocesses the data to remove noise and handle missing data fields. Then data transformation, such as the reduction of the number of variables and the discretization of attributes defined on a continuous domain, is often performed. The transformed data are provided to a data mining algorithm [1].

One of the important and complex issues in data mining is related to the transformation process. Kusiak [15] emphasized that the quality of knowledge discovery from a data set can be enhanced by transformation such as discretization because many of knowledge discovery techniques are very sensitive to the size of data in terms of complexity. In addition numerical value ranges are not easy enough for evaluation functions to handle in a nominal domain; for example, the original versions of the popular machine learning algorithms ID3 and AQ could be used only for categorical and symbolic data, and Quinlan [19] had to transform continuous ones into discrete values in his C4.5 decision tree learner. Many real-world classification algorithms are hard to solve unless the continuous attributes are discretized. It is hard to determine the intervals for a discretization of numerical attributes that has an infinite number of candidates. It remains a notorious problem in numerical attribute handling [8].

In the Top-Down Induction of Decision Trees family, the algorithms for discretization are based mostly on binarization within a subset of training data [3]. A simple discretization procedure divides the range of a continuous variable into equal-width intervals or equal-frequency intervals. A variant of the method uses Shannon's entropy theory so that the entropy scheme determines the interval boundary points [20]. Both Ching *et all.* [5] and Fayyad *et al.* [9] suggested class dependent algorithms. Those algorithms reduce the number of attributed values maintaining the relationship between the class and attribute values.

Liu *et al.* [16] classified discretization methods from five different viewpoints: *supervised* vs. *unsupervised*, *static* vs. *dynamic*, *global* vs. *local*, *top-down* vs. *bottom-up*, and *direct* vs. *incremental*. Unsupervised methods do not make use of class information in the discretization process while supervised methods utilize it. If no class information is available, unsupervised discretization is the only method possible. Dynamic methods perform discretization of continuous values during classification process, such as in C4.5 while static methods preprocess discretization before classification process. Local methods use the a local region of the instance space while global methods use the entire space. Top-down methods start with one interval and split intervals in the process of discretization. Bottom-up methods split completely all the continuous values of the attribute and merge intervals in the process of discretization. Direct methods divide the range into k intervals simultaneously, thus the number of intervals is provided. While incremental methods begin with a simple discretization and pass through an improvement process, needing an additional criteria which determines to stop discretizing.

Traditional interval discretization methods are based on a crisp set; a value in a continuous attribute must belong to only one interval. They are often not appropriate for describing a value located around the boundaries of intervals. Fuzzy logic is an attractive method for the representation of human knowledge in classification problems. A considerable number of studies have been conducted on the effects of fuzzy discretization on classification problems. Ishibuchi *et al.* [11, 13] examined the effect of fuzzy partitions on the performance of fuzzy rule-based systems and also discretized the domain of each attribute based on inhomogeneous fuzzy partitions in order to determine the optimal intervals.

An important decision in fuzzy partitioning is about determining the number of intervals, the positions of interval boundaries, and the degrees of overlapping in the fuzzy sets. Ishibuchi *et al.* [12] showed how fuzzy discretization can be derived from conventional interval discretization. Murata [17] adjusted fuzzy partitions by a genetic algorithm and Chen [4] proposed the concepts of interval-valued fuzzy hypergraph in order to generate fuzzy partitions.

In this paper, we optimize the parameters that specify fuzzy partitioning by genetic algorithms. We evolve the number of intervals, the boundaries, and the degrees of overlapping for fuzzy sets. For efficiency, we mix genes of discrete values and continuous values. We compare the classification accuracies of the suggested genetic fuzzy discretization methods with other discretization methods. For evaluation, we use artificial neural network and C4.5 as machine learning algorithms.

This paper is organized as follows. In Section 2, we briefly describe simple discretization methods and fuzzy discretization. Then we describe the main idea in Section 3. Section 4 explains datasets which are used in experiments and the experimental results. In Section 5, we make conclusions.

2. FRAMEWORK FOR FLEXIBLE DISCRETIZATION

2.1 Simple Discretization Methods

The representative discretization methods are equal-width discretization and equal-frequency discretization [6]. In the equal-width discretization, the whole range is divided into k intervals of equal width between v_{min} and v_{max}. The length w of each interval is $(v_{max} - v_{min})/k$. The bounds of the i^{th} interval are defined as

$$v[i]_{min} = v_{min} + w * i$$
$$v[i]_{max} = v_{min} + w * (i + 1)$$

where $i = 0, \ldots, k - 1$.

The membership function of a continuous attribute is defined as follows:

$$S_i(x) = \begin{cases} 1, & \text{if } v[i]_{min} \le x < v[i]_{max}, \\ 0, & \text{otherwise.} \end{cases}$$

In the equal-frequency discretization, it sorts the values of the attribute and divides them into k groups that each includes the same number of samples. Let N be the number of observed values of an attribute. The size of each interval is defined as $w = N/k$ .The bounds of each interval are defined as follows:

$$v[i]_{min} = (i * w)^{th} \ value$$
$$v[i]_{max} = ((i + 1) * w)^{th} \ value$$

where $i = 0, \ldots, k-1$. The membership function is defined as follows:

$$S_i(x) = \begin{cases} 1, & \text{if } v[i]_{min} \le r[x] < v[i]_{max}, i = 0, 1, ..., k - 2 \\ 1, & \text{if } v[i]_{min} \le r[x] \le v[i]_{max}, i = k - 1 \\ 0, & \text{otherwise.} \end{cases}$$

where $r[x]$ is the rank of value x in the sorted list.

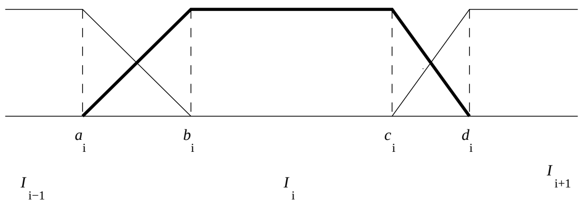

Figure 1: Trapezoidal membership function of a fuzzy set.

2.2 Fuzzy Discretization

A value in interval discretization methods must belong to only one interval without considering whether it has relevance to other adjacent intervals. For example, assume that the domain of our ages is divided into two intervals by the threshold age 25. Consider four ages, 11, 24, 26, and 39. Although 24 is more relevant to 26 than to 11, 11 and 24 belong to the same interval whereas 24 and 26 belong to different intervals. The relative difference of the values is not very well reflected. In this case, the simple discretization causes notable information loss.

A fuzzy set which does not require mutually exclusive intervals is adequate to resolve this problem. In some cases, more than one interval is associated with a value when it is located around a boundary between adjacent intervals. Fig. 1 shows a classical fuzzy interval I_i and its four parameters a_i, b_i, c_i, and d_i.

2.3 Fuzzy Discretization with GA

We divide the range of a continuous attribute into k intervals and represent each value by a k-bit string where each bit corresponds to one interval. The i^{th} bit of the binary string represents whether the value belongs to the i^{th} interval or not. While a value belongs to only one interval in a simple discretization, it can belong to more than one interval in fuzzy discretization. Thus a value can be represented by a binary mask. Fig. 2 shows an example. The value 0.523 belongs to the second and third intervals and is represented by a binary mask 0110. It provides more flexibility in machine learning algorithms for pattern classification. We optimize the boundaries of intervals and the degrees of overlapping in fuzzy discretization. We use four parameters for each interval I_i: t_i, t_{i+1}, l_i and u_i. The genetic fuzzy membership function is defined as follows:

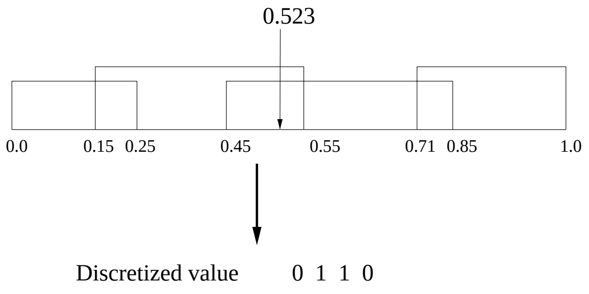

Figure 2: An example discretization and binary mask for a value with 4 intervals

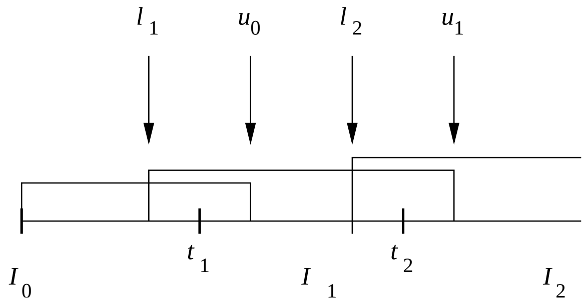

Figure 3: Parameters determining the boundaries of intervals and the degree of overlapping

$$G_0(a) = \begin{cases} 1, & \text{if } v_{min} \le a < u_0, \\ 0, & \text{otherwise.} \end{cases}$$

$$G_i(a) = \begin{cases} 1, & \text{if } l_i \le a < u_i, \quad i = 1, \ldots, k-2, \\ 0, & \text{otherwise.} \end{cases}$$

$$G_{k-1}(a) = \begin{cases} 1, & \text{if } l_{k-1} \le a \le v_{max}, \\ 0, & \text{otherwise.} \end{cases}$$

where $l_i \le t_i$, $i = 1, \ldots, k-1$, and $t_{i+1} \le u_i$, $i = 0, \ldots, k-2$. In the above, t_i and t_{i+1} indicate the "pivot" points of interval i. l_i and u_i indicates the extended left and right boundaries of the interval i, respectively. t_i, l_i and u_i are determined by genetic optimization (Fig. 3). Fig. 4 shows an example of discretization results. In the fuzzy discretization 1, the value 0.523 belongs to the second and third intervals and is represented by a mask 0110. In the fuzzy discretization 2, it belongs to only the third interval and is represented by 0010. The value 0.69 is represented by 0010 and 0011, respectively, in the two fuzzy discretization schemes. For each input of a neural network, we have four input nodes that accept four binary digits.

3. THE GA FOR GENETIC FUZZY DISCRETIZATION

3.1 Encoding

Fig. 6 shows an example chromosome. In the figure, we assume that the maximum number of intervals is 10 and the minimum is 3. The first nine genes indicate the validity of the boundaries of intervals, the next nine genes indicate the boundaries of the intervals, and the other eighteen genes

```
Create initial population;
Evaluate all chromosomes ;
**do**
{
    Choose parent1 and parent2 from population ;
    offspring = crossover (parent1, parent2) ;
    mutation(offspring) ;
    evaluation (offspring) ;
    replace(parent1, parent2, offspring);
} until (stopping condition) ;
return the best solution;
```

Figure 7: Genetic algorithm framework.

indicate the overlap degrees with adjacent intervals. In the case of maximum k-interval optimization, a chromosome has $4(k-1)$ genes. Among them, the first $k-1$ genes, *validity genes*, determine the validity of the boundaries. The next $k-1$ genes, *pivot genes*, correspond to the boundaries of the intervals and the other $2(k-1)$ genes, *overlap genes*, specify the degrees of overlapping on each adjacent intervals.

The validity genes activate or deactivate the pivot genes that determine the boundary points of intervals. When all validity genes have the value 1, the number of intervals is 10. When the k^{th} validity gene changes from 1 to 0, the k^{th} pivot gene is deactivated. It means merging I_k and I_{k+1} intervals. To the contrary, if the k^{th} validity gene changes from 0 to 1, the k^{th} pivot gene is activated and the k^{th} pivot point becomes a split point (Fig. 5).

Considering the volume of search space, we restrict the values of base genes to be discrete while allowing those of overlap genes to be real. We mix genes with discrete numbers and those with continuous numbers in a chromosome. The initial values of overlap genes are continuous numbers that are randomly chosen between 0.0 and 0.4. k was set to 10. The values of pivot genes $(t_1, \ldots, t_{k-1})$ are chosen from $\{0.10, 0.11, 0.12, 0.13, \cdots, 0.97, 0.98, 0.99\}$, and the gap between two adjacent valid boundaries is between 0.1 and 0.4. The value of an overlap gene indicates the distance between a boundary point and a pivot point. If the left overlap gene value for interval i's left boundary is 0.13, the pivot gene value of interval i is 0.5, and the width of interval i is 0.35, then the extended left boundary becomes 0.5 − 0.35 * 0.13 = 0.4545. The values of overlap genes should be in the range [0, 1).

3.2 Initialization

Initial solutions are generated at random. In each chromosome, the first nine genes are binary values, 0 or 1, which indicates the validity of corresponding pivot genes. The next nine genes are discrete values as mentioned above and the other eighteen genes are real values as mentioned above. We set the population size to 500.

3.3 Parent Selection

The fitness of a chromosome is determined by the accuracy of a machine learning algorithm on a predefined subset. We use the binary tournament selection [10].

Value	0.523	0.69	
Interval discretization	0 0 1 0	0 0 1 0	Even widths (interval width = 0.25)
Fuzzy discretization 1	0 1 1 0	0 0 1 0	
Fuzzy discretization 2	0 0 1 0	0 0 1 1	

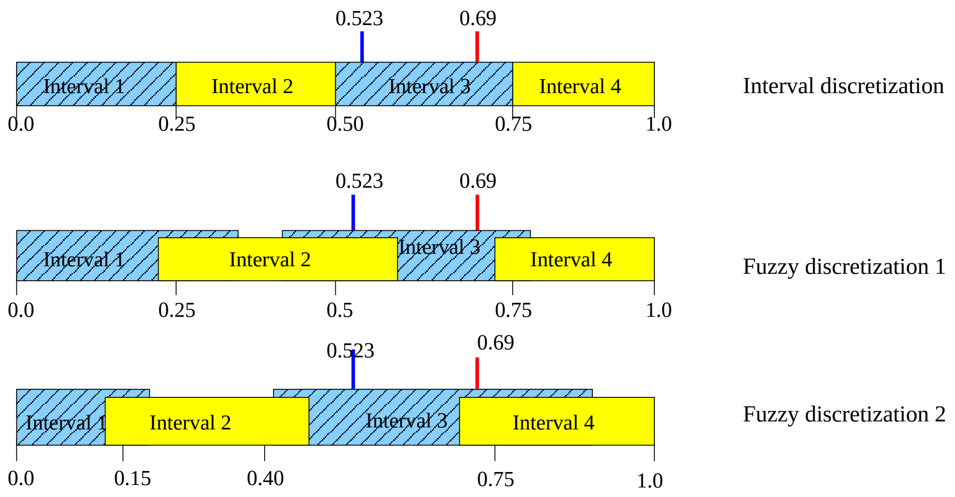

Figure 4: Three different example discretizations and the corresponding binary masks

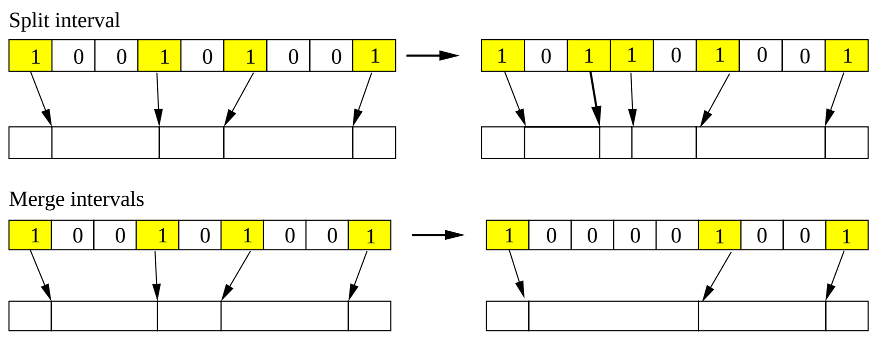

Figure 5: Merge and split intervals by validity genes

3.4 Crossover and Mutation

We use the uniform crossover [21]. Mutation operator assigns a random value to each overlap gene within its admitted range with a probability 0.3. We allow more conservative mutation to the base genes and each base gene is changed at random within a valid range with a probability 0.1.

3.5 Fitness Evaluation

Each solution is evaluated by ANN. To provide inputs for the ANN, each attribute value is transformed to k binary values that each represents a bit of the binary mask for the fuzzy membership where k is the number of intervals. The neural networks are trained by a set of training data. For example, if there are 21 attributes and 3 classes in a data mining problem as in Section 4, each neural network has 84 input nodes and 3 output nodes when the number of intervals is 4. The number of hidden nodes is determined to be $\lfloor\sqrt{84}\rfloor$.

3.6 Replacement

The offspring replaces a solution in the population by the following rule: the more similar parent to the offspring is replaced if the offspring is better; otherwise, the other parent is replaced if the offspring is better; if not again, the worst chromosome in the population is replaced [2].

3.7 Stopping Condition

The GA stops after a fixed number (100,000) of generations, or after a given number (5,000) of consecutive failures in replacing one of the parents.

4. EXPERIMENTAL RESULTS

4.1 Database

We used two well known datasets from the UCI Machine Learning Database Repository[1] to measure the performance of the proposed approach. The statistics are outlined in Table 1. The WDG contains 5,000 instances and each instance

[1]http://www.ics.uci.edu/~mlearn/MLRepository.html

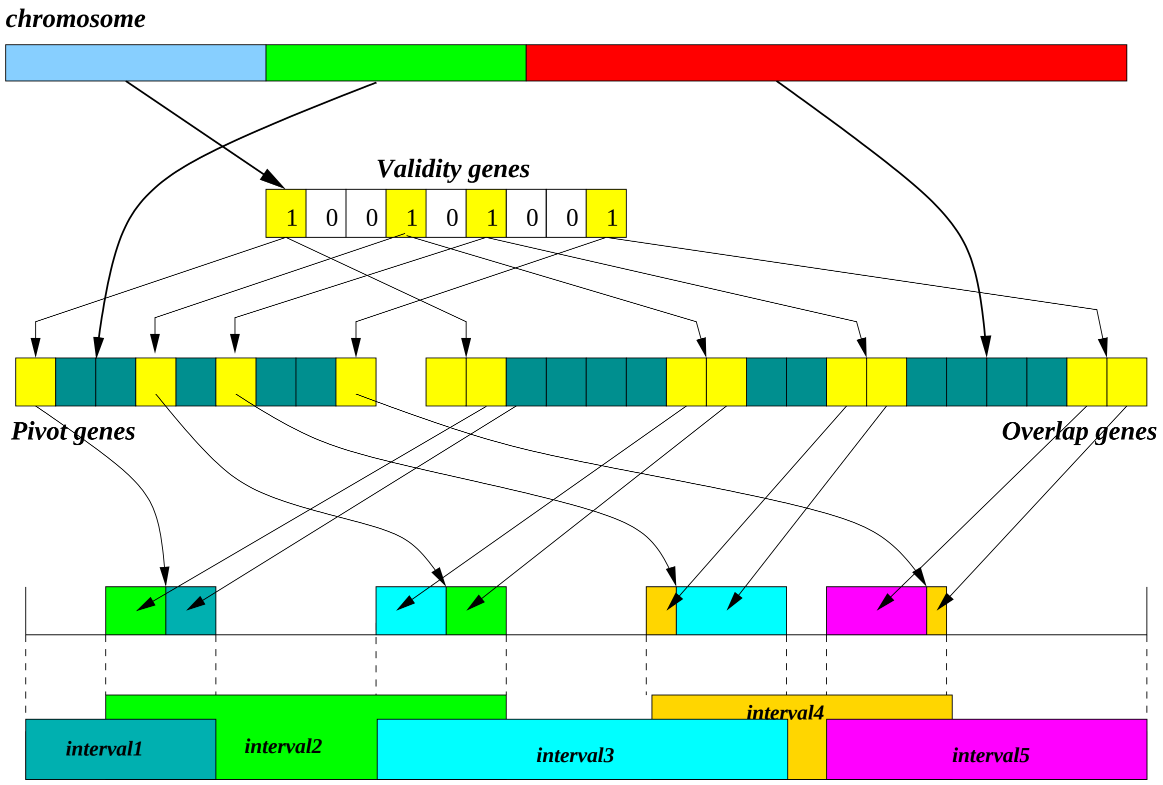

Figure 6: Chromosome with 36 genes for various interval partitioning.

Table 1: Datasets

Database	# of Instances	# of Attributes	# of Classes
WDG[†]	5,000	21	3
WDBC[‡]	569	30	2

† Waveform Database Generator.
‡ Wisconsin Diagnostic Breast Cancer.

belongs to one of three classes of waves. Each class has approximately the same number of instances. Each instance has 21 attributes with continuous values between 0 and 6 and all attributes include noise. The WDBC contains 569 instances with 357 benign and 212 malignant cases. Each instance is described by an index, diagnosis, and 30 real-valued attributes. As a preprocessing procedure for genetic fuzzy discretization, we normalized each attribute value to lie in the range [0, 1]. The domain of each attribute is discretized into five intervals. For robust comparison, we applied the n-fold cross-validation [7, 14]. That is, we randomly divide the entire data set D into n mutually exclusive, equal-sized subsets $D_1, D_2, \ldots$, and, D_n. The i^{th} experiment was trained with $D \backslash D_i$ and tested with D_i for $i = 1, \ldots, n$. For the WDG we set n to 3 but to 2 for the WDBC because of the small number of instances.

4.2 Experimental Results

We compared three discretization methods. All programs were written in $C++$ language and run on AMD Athlon XP 1.8 GHz under the Linux operating system. In order to evaluate the generated data sets, we use artificial neural network and C4.5. C4.5 [18] is a typical top-down method for inducing decision trees. The simulation results are summarized in Table 2 and Table 3. “ID” indicates the discretization method that divides the entire domain into mutually exclusive intervals evenly based on the pre-defined number, k, of intervals. “FD” indicates a conventional fuzzy discretization with a pre-specified overlap degree. “GAFD” indicates the suggested genetic fuzzy discretization that evolves the number,k, of intervals, the widths of intervals, and the degrees of overlaps. In the tables, we show the average accuracies and the standard deviations (σ) over 1000 runs. $\sigma/\sqrt{n}$ represents the group standard deviation of n runs of ANN.

For ID and FD, we tested with each k of 3 through 10 and took the best one in the Tables 2 and 3. Fig. 8 and Fig. 9 show the details of the results which are not shown in the tables. The results show that a simple increase in the number of intervals did not always make good accuracies (Fig. 8 and Fig. 9). From the simulation results, we can observe that the GAFD improved the classification accuracies over ID and FD under both the frameworks of ANN and C4.5. Considering the group standard deviation in the case of ANN, we can say that GAFD was better than the others with a confidence of more than 99%.

Table 2: Result on the dataset: Wisconsin Diagnostic Breast Cancer

Method	ANN (n=1000)			C4.5	
	Accuracy Avg(%)	$\sigma/\sqrt{n}$	Intervals(k)	Accuracy(%)	Intervals(k)
ID	94.64764	0.03802	4	93.14554	6
FD	94.86219	0.03189	5	93.49951	7
GAFD	**96.42861**	0.03134	4	**93.67433**	4

Table 3: Result on the dataset: Waveform Database Generator

Method	ANN (n=1000)			C4.5	
	Accuracy Avg(%)	$\sigma/\sqrt{n}$	Intervals(k)	Accuracy(%)	Intervals(k)
ID	82.35543	0.03917	8	73.78524	5
FD	82.84607	0.03648	6	74.16517	6
GAFD	**83.14135**	0.03550	7	**75.84483**	7

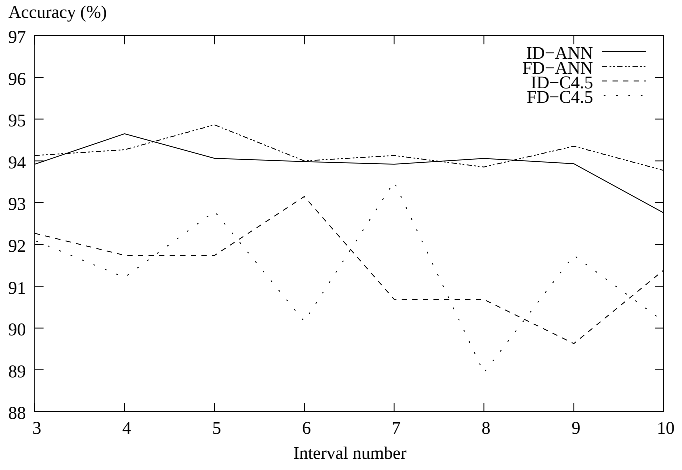

Figure 8: Accuracies with various interval partitions (WDBC).

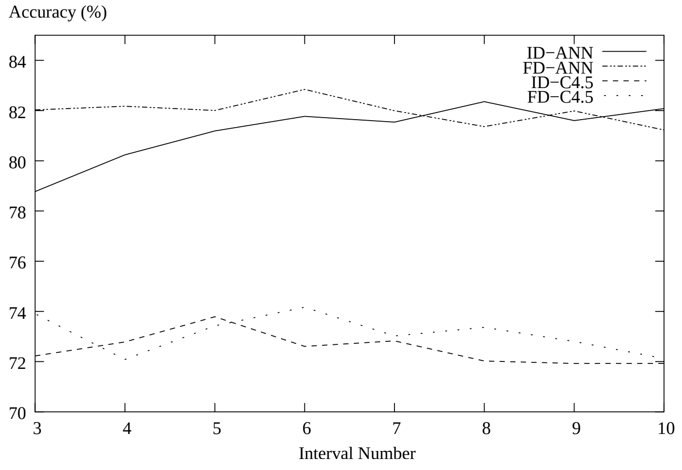

Figure 9: Accuracies with various interval partitions (WDG).

5. CONCLUSIONS

In this paper, we tried to optimize with a genetic algorithm the number of intervals, the size of intervals, and the overlap degrees that specify the fuzzy membership function to find a more accurate discretization method. Our approach is static, supervised method. In a learning process we determine the parameters of discretization such as the number of intervals, the sizes of intervals, and the overlap degrees of fuzzy membership function and all the continuous values of attributes are discretized. Considerable improvement was observed by the suggested genetic fuzzy discretization. In the result we observed that the number of intervals did not have the key relevance to the performance of the classification methods and the positions of pivot points which determine the boundaries of intervals were more relevant to the accuracy of the methods [22].

6. ACKNOWLEDGMENTS

This work was supported by the Brain Korea 21 Project. The ICT at Seoul National University provided research facilities for this study.

7. REFERENCES

[1] P. Berka and I. Bruha. Discretization and grouping: Preprocessing steps for data mining. In *Principles of Data Mining and Knowledge Discovery*, pages 239–245, 1998.

[2] T. N. Bui and B. R. Moon. Genetic algorithm and graph partitioning. *IEEE Trans. on Computers*, 45(7):841–855, 1996.

[3] J. Catlett. On changing continuous attributes into ordered discrete attributes. In *Proceedings of the European Working Session on Learning*, pages 164–178., 1991.

[4] S. M. Chen. Interval-valued fuzzy hypergraph and fuzzy partition. *IEEE Trans. on Systems, Man and Cybernetics, Part B*, 27(4):725–733, 1997.

[5] J. Y. Ching, A. K. C. Wong, and K. C. C. Chan. Class-dependent discretization for inductive learning from continuous and mixed mode data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(7):641–651, 1995.

[6] David K. Y. Chiu, Andrew K. C. Wong, and B. Cheung. Information discovery through hierarchical maximum entropy discretization and synthesis. In

Knowledge Discovery in Databases, pages 125–140. 1991.

[7] B. Efron and R.Tibshirani. Cross-validation and the bootstrap: Estimating the error rate of a prediction rule. In *Technical Report(TR-477), Dept. of Statistics, Stanford University.*, 1995.

[8] T. Elomaa and J. Rousu. General and efficient multisplitting of numerical attributes. *Machine Learing*, 36(3):201–244, 1999.

[9] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous attributes as preprocessing for classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann*, pages 1022–1027, 1995.

[10] D. E. Goldberg, K. Deb, and B. Korb. Do not worry, be messy. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 24–30, 1991.

[11] H. Ishibuchi and T. Yamamoto. Fuzzy rule selection by data mining criteria and genetic algorithms. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 399–406, 2002.

[12] H. Ishibuchi and T. Yamamoto. Deriving fuzzy discretization from interval discretization. In *Proc. of 2003 IEEE International Conference on Fuzzy Systems*, pages 749–754, 2003.

[13] H. Ishibuchi, T. Yamamoto, and T. Nakashima. Fuzzy data mining: effect of fuzzy discretization. In *Proc. of 2001 IEEE International Conference on Data Mining*, pages 241–248, 2001.

[14] R. Kohavi. A study of cross-validation and bootstrap of accuracy estimation and model selection. In *Proceedings of hte Rourteenth International Joint Conference on Artificial Intelligence*, pages 1137–1143, 1995.

[15] A. Kusiak. Feature transformation methods in data mining. *IEEE Trans. on Electronics Packaging Manufacturing*, 24(3):214–221, 2001.

[16] H. Liu, F. Hussain, C. L. Tan, and M. Dash. Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–423, 2002.

[17] T. Murata, H. Ishibuchi, and M. Gen. Adjusting fuzzy partitions by genetic algorithms and histograms for pattern classification problems. In *Proc. of 1998 IEEE International Conference on Evolutionary Computation*, pages 9 –14, 1998.

[18] J. R. Quinlan. C4.5: Programs for machine learning. *Morgan Kaufman*, 1993.

[19] J. R. Quinlan. Improved use of continuous atrributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77–90, 1996.

[20] D. W. Stashuk and R. K. Naphan. Probabilistic inference-based classification applied to myoelectric signal decomposition. *IEEE Transactions on Biomedical Engineering*, 39(4):346–355, 1992.

[21] G. Syswerda. Uniform crossover in genetic algorithms. In *International Conference on Genetic Algorithms*, pages 2–9, 1989.

[22] S. Trautzsch and P. Perner. A comparision of different multi-interval discretization methods for decision tree learning.