Using genetic algorithms in K-means fast learning artificial neural networks for clustering

Yin, Xiang

2006

Yin, X. (2006). Using genetic algorithms in K-means fast learning artificial neural networks for clustering. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/2560>

<https://doi.org/10.32657/10356/2560>

Nanyang Technological University

Downloaded on 26 Apr 2023 23:29:14 SGT

**USING GENETIC ALGORITHMS IN K-MEANS FAST
LEARNING ARTIFICIAL NEURAL NETWORKS FOR
CLUSTERING**

YIN XIANG

School of Computer Engineering

2006

Using Genetic Algorithms in K-Means Fast Learning Artificial Neural Networks for Clustering

Yin Xiang

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Master of Engineering

2006

Abstract

Vast amounts of data with various types of information are churned out each day, but it is often difficult to make effective use of the data in a meaningful way because the embodied knowledge and information is not completely understood. The K-means Fast Learning Artificial Neural Network (KFLANN) is a small neural network bearing two types of parameters, the tolerance, δ and the vigilance, ρ . It was shown in this research that the KFLANN is capable of fast and accurate assimilation of data. However, it was an unresolved issue to determine the suitable values for δ and ρ . This research project studies the KFLANN in depth and introduces Genetic Algorithms (GAs) as a possible solution for searching through the parameter space to effectively and efficiently extract suitable values to δ and ρ . It is also able to determine significant features of the data that help achieve accurate clustering. A detailed analysis of KFLANN and its hybridization with a Genetic Algorithm is shown in this report. Experimental results are presented to illustrate the hybrid GA-KFLANN ability using available test data. The comparison of the experimental results with other clustering techniques showed that introducing GA into KFLANN improves the overall performance of the original KFLANN algorithm and is able to find the suitable settings for tolerance and vigilance values.

Acknowledgement

I would like to take this opportunity to express my sincere appreciation to the following people who have in one way or another helped me through my master program:

- Dr. Tay Leng Phuan, Alex, my supervisor, for his invaluable guidance and enthusiastic support in accomplishment of the research and the thesis.
- Dr. Tan Teng Hong, Associate Consultant of Cardiology from KK Women's and Children's Hospital, for his enthusiastic help and guidance in explaining and providing useful information on medical data.
- Ms. Wong Lai Ping who has shared her opinions and given me suggestions on my project.
- Mr. Tan Swee Huat, Technical Staff of Center for Computational Intelligence, for his support to provide facilities and development environment.

Table of Contents

Abstract.....	1
Acknowledgement.....	4
Table of Contents	5
List of Tables	9
List of Figures	10
List of Equations	13
Chapter 1 Introduction	14
1.1 Motivations.....	14
1.2 Objectives	15
1.3 Thesis Outline	15
Chapter 2 Data Mining.....	18
2.1 Cluster Analysis.....	19
2.1.1 Structure of the Clustering System	19
2.2 The Curse of Dimensionality.....	20
2.3 Feature Selection.....	21
2.3.1 Selection Criteria	21
2.3.2 Search Procedures.....	22
2.4 Principal Component Analysis (PCA)	26
Chapter 3 Neural Networks	29
3.1 Introduction	29
3.2 Network Architectures	31

3.2.1	Single-Layer Feedforward Networks.....	31
3.2.2	Multilayer Feedforward Networks.....	32
3.2.3	Recurrent Networks	33
Chapter 4	Genetic Algorithms	35
4.1	Introduction	35
4.2	GAs for Optimization.....	36
4.2.1	Introduction to Optimization	36
4.2.2	Robustness of Conventional Search Techniques	37
Chapter 5	Hybridization	39
5.1	GA-Clustering.....	39
5.2	GA-DistAl	40
5.3	GA-RBF	41
Chapter 6	K-Means Fast Learning Artificial Neural Network.....	42
6.1	Architecture of KFLANN.....	42
6.2	Algorithm of KFLANN.....	43
6.3	Data Pattern Reshuffling	45
6.4	Network Parameter Settings	47
6.4.1	Tolerance Value	47
6.4.2	Vigilance Value	52
6.5	Feature Selection.....	52
6.6	KFLANN using Standard Deviation (SD-KFLANN)	53
6.7	KFLANN using Exhaustive Search (ES-KFLANN)	54

Chapter 7	Hybrid Model of GA-KFLANN	56
7.1	Chromosome Encoding	56
7.2	Population Initialization	57
7.3	Fitness Evaluation.....	58
7.4	Reproduction.....	59
7.5	Crossover	61
7.6	Mutation	62
7.7	Population Replacement.....	63
7.8	Crossover Hill-Climbing (XHC)	64
Chapter 8	Experiments and Results	66
8.1	Iris Data Set	66
8.1.1	Performance of GA-KFLANN with Non-Overlapping and Overlapping Generations.....	67
8.1.2	Performance of GA-KFLANN with and without XHC	68
8.1.3	Performance of GA-KFLANN with Uniform Mutation and Dynamic Mutation	70
8.1.4	Performance of GA-KFLANN with Different Mutation Probabilities (P_m)	72
8.1.5	Performance of GA-KFLANN with Different Crossover Probabilities (P_c)	74
8.1.6	Comparison with Other Methods.....	75
8.2	Wine Data.....	76
8.2.1	Performance of GA-KFLANN with Different Mutation Probabilities (P_m)	77
8.2.2	Performance of GA-KFLANN with Different Crossover Probabilities (P_c)	78

8.2.3	Comparison with Other Methods.....	79
8.3	Wisconsin Breast Cancer (WBC) Data	79
8.3.1	Performance of GA-KFLANN with Different Mutation Probabilities	80
8.3.2	Performance of GA-KFLANN with Different Crossover Probabilities	81
8.3.3	Comparison with Other Methods.....	82
8.4	Glass Data.....	83
8.4.1	Performance of GA-KFLANN with Different Mutation Probabilities	83
8.4.2	Performance of GA-KFLANN with Different Crossover Probabilities	84
8.4.3	Comparison with Other Methods.....	85
8.5	Pima Indians Diabetes Data.....	86
8.5.1	Performance of GA-KFLANN with Different Mutation Probabilities	87
8.5.2	Performance of GA-KFLANN with Different Crossover Probabilities	87
8.5.3	Comparison with Other Methods.....	88
8.6	Comparison with Exhaustive Search Algorithm	89
8.7	Overall Comparison	90
Chapter 9	Conclusions	92
Reference		95
Appendix A		100

List of Tables

Table 6.1 Experimental Results of SD-KFLANN on the Three Data Sets	53
Table 6.2 Experimental Results of ES-KFLANN on the Three Data Sets.....	54
Table 6.3 Comparison of Clustering Accuracy Between ES-KFLANN and K-NN.....	54
Table 7.1 Sample Chromosomes and Fitness values	60
Table 7.2 An Example of Uniform Crossover	62
Table 8.1 Comparison of Clustering Results on Iris Data.....	76
Table 8.2 Comparison of Clustering Results on Wine Data.....	79
Table 8.3 Comparison of Clustering Results on Wisconsin Breast Cancer Data	82
Table 8.4 Comparison of Clustering Results on Glass Data	86
Table 8.5 Comparison of Clustering Results on Pima Indians Diabetes Data	89
Table 8.6 Comparison of GA-KFLANN and ES-KFLANN on Feature Selection	89
Table 8.7 Comparison of GA-KFLANN and ES-KFLANN on Clustering Accuracy.....	89
Table 8.8 Comparison of GA-KFLANN and ES-KFLANN on Completion Time	89
Table 8.9 Comparison between GA-KFLANN and other Algorithms	90

List of Figures

Figure 2.1	The Structure of the Data Mining System	20
Figure 2.2	Search Tree for Feature Subsets.....	23
Figure 2.3	Sequential Search Techniques: (a) Forward Selection, (b) Backward Elimination	24
Figure 2.4	Data with Heavy Overlapping on Each Feature Alone	25
Figure 2.5	Illustration of PCA Applied to Data in Two Dimensions.....	27
Figure 3.1	Biological Neuron	29
Figure 3.2	A Simple Artificial Neuron.....	31
Figure 3.3	Single-Layer Feedforward Network.....	32
Figure 3.4	Multilayer Feedforward Network.....	33
Figure 3.5	Recurrent Network with Hidden Neuron.....	34
Figure 4.1	The GA process	36
Figure 6.1	Architecture of KFLANN	43
Figure 6.2	Different clusters are formed for the same data set when δ is varied.....	48
Figure 6.3	Gaussian Distribution Curve	49
Figure 7.1	The Architecture of GA-KFLANN	56
Figure 7.2	Sample Chromosome Encoding for the Iris Dataset	57
Figure 7.3	Roulette Wheel of SUS.....	61
Figure 8.1	Performance of GA-KFLANN with Non-Overlapping and Overlapping Generations	67
Figure 8.2	Plot of Max Accuracy vs. Generation with and without XHC	69

Figure 8.3	Plot of Average Accuracy vs. Generation with and without XHC	69
Figure 8.4	Plot of Max Accuracy vs. Generation with Different Mutation Methods	71
Figure 8.5	Plot of Average Accuracy vs. Generation with Different Mutation Methods	71
Figure 8.6	Plots of Average Accuracy Obtained Using Different Mutation Probabilities	72
Figure 8.7	Plots of Max Accuracy Obtained Using Different Mutation Probabilities....	73
Figure 8.8	Plots of Average Accuracy Obtained Using Different Crossover Probabilities	74
Figure 8.9	Plots of Max Accuracy Obtained Using Different Crossover Probabilities ..	74
Figure 8.10	Plots of Average Accuracy Obtained Using Different Mutation Probabilities	77
Figure 8.11	Plots of Average Accuracy Obtained Using Different Crossover Probabilities	78
Figure 8.12	Plots of Average Accuracy Obtained Using Different Mutation Probabilities	80
Figure 8.13	Plots of Average Accuracy Obtained Using Different Crossover Probabilities	81
Figure 8.14	Plots of Average Accuracy Obtained Using Different Mutation Probabilities	84
Figure 8.15	Plots of Average Accuracy Obtained Using Different Crossover Probabilities	85

Figure 8.16 Plots of Average Accuracy Obtained Using Different Mutation Probabilities	87
Figure 8.17 Plots of Average Accuracy Obtained Using Different Crossover Probabilities	88
Figure 8.18 Clustering Accuracy Obtained Using GA-KFLANN, SD-KFLANN, and K-NN	91

List of Equations

Equation 2.1	22
Equation 2.2	26
Equation 2.3	26
Equation 2.4	26
Equation 3.1	30
Equation 3.2	30
Equation 6.1	44
Equation 6.2	44
Equation 6.3	50
Equation 6.4	52
Equation 7.1	58
Equation 7.2	59
Equation 7.3	62
Equation 7.4	62
Equation 7.5	63

Chapter 1 Introduction

1.1 Motivations

As companies begin to rely heavily on information technology, databases in various fields including science, technology, production, education and business become increasingly huge. Within the huge mass of data, lies the possibility of strategically important information. It is however not a trivial task to extract such meaningful information from the huge amount of data. Data mining is being used in organizations worldwide to understand trends and behaviours of high-value customers so that the company's product and sales techniques can be reconfigured to maximize benefits to the company. Sometimes it is important to cluster data of different properties into categories in order to analyze their behaviours. For example, customers from different age groups can have different buying activities for some products.

The Artificial Neural Network (ANN) is a well-known technique in computational intelligence and it can be used to solve data clustering tasks. Based on the effort of many researchers from many fields of study including biological, psychological, computer science, and mathematical areas, various models of neural networks capable of extracting patterns have been developed. Some of these trends and patterns hidden inside data are too complex to be noticed either by human or other computer techniques. The Fast Learning Artificial Neural Network (FLANN) is a neural network developed with hybridized properties of Kohonen Network and the Adaptive Resonance Theory (ART)

architectures [22],[31],[36]. KFLANN enforces FLANN with the K-means calculation in order to make consistent centroid formations regardless of data presentation sequence. A variety of techniques have been developed to obtain suitable network parameters of KFLANN for efficient and effective clustering as well as feature extraction. However, it is difficult for these techniques to obtain optimal parameters and the search durations are extremely long. The idea is to adapt KFLANN for efficient and effective data mining.

1.2 Objectives

The objectives of this work are as follows:

- To study the characteristics of KFLANN so as to understand how its network parameters affect the performance of the neural network in data mining
- To investigate various optimization techniques and decide the most suitable one to hybridize with KFLANN
- To develop the hybridization of Genetic Algorithm and KFLANN
- To test the performance of the hybrid model using a few well-known data sets and compare its clustering results with those from other clustering techniques

1.3 Thesis Outline

This thesis consists of nine chapters. The initial six chapters cover the ground work needed in this project and are available in published literature and the last three chapters reflect the original contributions of this project. Chapter 1 gives a brief introduction of the background and organization of this report.

Chapter 2 introduces the basic concepts of data mining and overviews the structure of the data mining system to be investigated. Several dimension reduction methods are also described in this chapter.

Chapter 3 gives an overview of artificial neural networks and introduces various types of neural network architectures.

Chapter 4 provides basic ideas of Genetic Algorithms (GA) and explains why GAs are suitable for complex optimization problem such as parameter search of KFLANN.

Chapter 5 explains the idea of hybridization of GAs and ANNs and its benefit. Related works in this area are also briefly introduced in this chapter.

Chapter 6 provides an introduction to KFLANN algorithm as well as detailed network parameter analysis. The problem to be solved in this research is also explained in this chapter.

Chapter 7 proposes a hybrid GA-KFLANN. The architecture and parameter setting of this hybrid model are discussed in the chapter.

Chapter 8 presents experimental results of GA-KFLANN on a few real life data sets. Comparisons of these results are made among the original KFLANN, K-NN and GA-KFLANN in order to illustrate the feasibility and performance of GA-KFLANN.

Chapter 9 concludes this research work.

Chapter 2 Data Mining

In recent years, the information industry has come to realize that useful information and knowledge can be obtained from the huge storages of data. The information and knowledge gained can be used for applications ranging from business management, production control, market analysis, engineering design, and science exploration [16]. However, the fast-growing collection of data, with its numerous dimensions, has far exceeded the human ability for comprehension. Without powerful tools, it is impossible to realize the task of extracting valuable gems from the pile.

Data Mining is an analytic process used to explore data to search for consistent patterns or systematic relationships between variables [33]. Data mining performs the following functions.

- Concept/class description in summarized, concise, and yet precise terms through data characterization or data discrimination.
- Association analysis to discover rules showing attribute-value conditions that occur frequently together in a given set of data.
- Classification and prediction to find a set of models that describe and distinguish classes or concepts from a set of training data, to use the model to predict the class of objects whose class label is unknown.

- Cluster analysis to analyze data objects without consulting a known class label based on the principle of maximizing the intra-class similarity and minimizing the interclass similarity.
- Outlier analysis to analyze data objects that do not comply with the general behaviour or model of the data.
- Evolution analysis to describe and model regularities or trends for objects whose behaviour changes over time.

2.1 Cluster Analysis

Clustering is the process of grouping data into classes or clusters so that objects within a cluster have high similarity in comparison to one another, but dissimilar to objects in other clusters. Cluster analysis has been widely used in numerous applications, including pattern recognition, data analysis, image processing, and market research. By clustering, one can identify dense and sparse regions and, therefore, discover overall distribution patterns and interesting correlations among data attributes [16].

2.1.1 Structure of the Clustering System

The structure of the clustering system to be addressed in this thesis is shown in Figure 2.1.

It consists of the following steps:

- Raw data set will be preprocessed if it is not well structured.
- K-FLANN is employed to read the processed data set and classification is performed based on its tolerance and vigilance values.

- Significance analysis is conducted between resulting classes of K-FLANN and the desired class assignments of the data set.
- Accurate classifications and the corresponding tolerance and vigilance values of K-FLANN are recorded and saved in files.

Figure 2.1 The Structure of the Data Mining System

2.2 The Curse of Dimensionality

In practice, real life data is complex, consisting of numerous features that are used to distinguish the differences amongst patterns. For example, in a cardiac database recording symptoms and diagnosis of patients, there can be about one hundred features including atria septum, ventricle septum, ventricle contractility, tricuspid annulus, aorta,

and so on. However, the large number of features can greatly reduce the performance and increase the computation of a data mining system. Consider a data set with d features. If each feature is to be divided into M smaller intervals to improve the precision of a data mining procedure, the total number of possible pattern combinations increases to M^d , leading to an exponential increase in the computational requirements. This phenomenon has been termed the *curse of dimensionality* [4], [29].

2.3 Feature Selection

A common principle in dimension reduction is to select a subset of the inputs that are more significant within the problem space and discard the remainder with lesser significance. This approach can be useful if many of the inputs carry little useful information for the resolution of the problem, or if sets of inputs bear strong inter-feature correlations such that the same information is repeated over several variables [4].

2.3.1 Selection Criteria

For the classification problem, the selection criteria should ideally be taken as the probability of misclassification. As the number of features is reduced, the generalization performance of the system will improve until some optimal subset of features is reached. If fewer than necessary features are retained the performance will degrade. One limitation of selection criteria, such as those based on class separability, is the inability of modeling the phenomenon. In general, such measures J in Equation 2.1 satisfy a monotonicity property such that

$$J(X+) \geq J(X)$$

Equation 2.1

where X denotes a set of features, and $X+$ denotes a larger set of features which contains the set X as a subset, J denotes the selection criteria, such as the classification accuracy.

The inequality simply states that the error rate cannot be reduced by merely deleting features. As a consequence, criteria that satisfy the monotonicity constraint cannot be used to determine the optimum size for a set of variables and thus cannot be used to compare sets of different sizes. However, it offers a useful way to compare sets of variables having the same number of elements. Conventional statistical approach can address this issue and is used to measure the significance of the improvement in discrimination resulting from inclusion of extra variables [10]. Another approach is to apply cross-validation techniques to compare models trained using different numbers of features [2].

2.3.2 Search Procedures

The search procedure to locate the ideal subset of input features can be time consuming and expensive. If there are d features, then since each feature can be present or absent, there are a total of 2^d possible feature subsets which could be considered. For a relatively small number of features an exhaustive search can be considered, while for large numbers of input variables, exhaustive search becomes prohibitively expensive.

2.3.2.1 Exhaustive Search

The exhaustive search technique basically tests all possible combinations of features on selection criteria. Consider a set of 5 features indexed by the labels 1, 2, 3, 4, 5 and a subset of size 2 is expected. The search tree shown in Figure 2.2 can be constructed. The number next to each node denotes the feature which is eliminated at that node. Each possible subset of two features selection from a total of five is represented in the bracket at the bottom of the tree.

Figure 2.2 Search Tree for Feature Subsets

The exhaustive search is in general the only approach which is guaranteed to find the optimal subset. In practical application, however, it is forced to consider simplified selection criteria as well as non-exhaustive search procedures in order to limit the computational complexity.

2.3.2.2 Branch and Bound Techniques

Branch and bound is an accelerated search procedure using criteria which satisfy the monotonicity relation shown in Equation 2.1 and it is guaranteed to find the best subset of a given size without having to evaluate all possible subsets [26]. Suppose a criterion $J(d-)$ is to be maximized and the value of J corresponding to the node shown at A in Figure 2.2 is recorded as a threshold. If at any point in the search an intermediate node is

encountered, such as B, for which the value of J is smaller than the threshold, the nodes shown as solid circles need not be evaluated. If at any point in the search a final-layer node is encountered which has a larger value for the criterion, then this value becomes the new threshold. The algorithm terminates when every final-layer node has either been evaluated or excluded using the monotonicity relation.

2.3.2.3 Sequential Search Techniques

The branch and bound algorithm for monotonic selection criteria is generally faster than exhaustive search but is computationally expensive. Sometimes significantly faster techniques giving suboptimal solutions are more appropriate and meaningful. The simplest method would be to select d features that are individually the best (obtained by evaluating the selection criteria using one feature a time). This method, however, is likely to be highly unreliable and would only be optimal for selection criteria which can be expressed as the sum or product of the criteria evaluated for each feature individually, and it would therefore only be appropriate if the features were completely independent.

Figure 2.3 Sequential Search Techniques: (a) Forward Selection, (b) Backward Elimination

A better approach, known as sequential forward selection, is illustrated in Figure 2.3(a).

The procedure begins by considering each of the variables individually and selecting the

one which gives the largest value for the selection criterion. At each successive stage of the algorithm, one additional feature is added to the set, again selected base on its bearing towards the largest increase in the value of the selection criterion. One obvious difficulty with this approach is that, if there are two features of the kind shown in Figure 2.4, such that either feature alone provides little discrimination, but where both features together are very effective, then the forward selection procedure may never find this combination since either feature alone would never be selected.

Figure 2.4 Data with Heavy Overlapping on Each Feature Alone

An alternative is to start with the full set of d features and to eliminate them one at a time. This gives rise to the technique of sequential backward elimination illustrated in Figure 2.3(b). At each stage of the algorithm, one feature is deleted from the set, chosen from amongst all available candidates as the one which gives the smallest reduction in the value of the selection criterion. This overcomes the problem with the forward selection approach but requires a greater number of evaluations and is still not guaranteed to be optimal.

2.4 Principal Component Analysis (PCA)

PCA is a statistical technique that linearly transforms an original set of variables into a substantially smaller set of uncorrelated variables that represents most of the information in the original set of variables. Its goal is to reduce the dimensionality of the original data set [12].

Suppose that the goal is to map vectors $\mathbf{x}^n$ in a d -dimensional space $(x_1, \dots, x_d)$ onto vectors $\mathbf{z}^n$ in an M -dimensional space $(z_1, \dots, z_M)$, where $M < d$. The vector $\mathbf{x}$ can be approximated by a linear expression of a set of d orthonormal vectors $\mathbf{u}_i$ and constants b_i as shown in Equation 2.2

$$\tilde{\mathbf{x}} = \sum_{i=1}^M z_i \mathbf{u}_i + \sum_{i=M+1}^d b_i \mathbf{u}_i \quad \text{Equation 2.2}$$

The basis vectors $\mathbf{u}_i$ satisfies Equation 2.3 and the detailed derivation can be found in [13]

$$C\mathbf{u}_i = \lambda_i \mathbf{u}_i \quad \text{Equation 2.3}$$

where C is the covariance matrix of the set of vectors $\mathbf{x}^n$ and is given by Equation 2.4

$$C = \sum_n (\mathbf{x}^n - \bar{\mathbf{x}})(\mathbf{x}^n - \bar{\mathbf{x}})^T \quad \text{Equation 2.4}$$

The basis vectors $\mathbf{u}_i$ are thus the eigenvectors of the covariance matrix. The linear dimensionality reduction procedure is called the *Karhunen-Loève transformation* or *principal component analysis*. Each of the eigenvectors $\mathbf{u}_i$ is called a *principal component*. The technique is illustrated in Figure 2.5 for the case of data points in two dimensions. [4]

Figure 2.5 Illustration of PCA Applied to Data in Two Dimensions

Two principal components are extracted from the data in two dimensions: u_1 and u_2 . Figure 2.5 illustrates why the principal component u_1 contains most of the information, and thereby data projected on the new dimension of u_1 have a larger variance comparing with those on u_2 . Therefore, u_1 alone is able to represent the data with just 1 dimension.

However, PCA is limited by virtue of being a linear technique. It may therefore be unable to capture more complex nonlinear correlations, and may therefore overestimate the true dimensionality of the data [4]. Moreover, principal components are transformed variables which do not have intuitive meaning and lose some information of the original variables during the projection onto new axes. In data mining, further analysis of the original variables can sometimes be important, since the principal components lack this information, it is not as useful in some cases.

In this research an important objective is to conduct feature extraction on data sets and analyze the significance of different attributes. Therefore, PCA was not used as a tool of dimension reduction.

Chapter 3 Neural Networks

3.1 Introduction

A biological neural network acts as a massively parallel-distributed processor made up of many interconnected neurons that have natural propensity for storing experiential knowledge and making it available for future use. The network acquires knowledge from its environment through a learning process and interneuron connection strengths, known as synaptic weights, which are used to store the acquired knowledge. During the learning process, the network constantly modifies the synaptic weights to achieve the best outcome with the minimum error using the acquired knowledge [32].

A neuron is the basic computation or information-processing unit of the brain. Figure 3.1 taken from [32] shows the basic components of a biological neuron.

Figure 3.1 Biological Neuron

In the human brain, a typical neuron collects signals from other neurons through a host of fine structures called *dendrites*. When it receives excitatory input that is sufficiently large compared with its inhibitory input, the neuron sends out spikes of electrical activity through a long, thin strand known as an *axon*, which splits into thousands of branches. At the end of each branch, a structure called a *synapse* converts the activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes [6].

An artificial neural network is a mathematical model of a biological neural network, which simulates the brain's action. A simple artificial neuron is illustrated in Figure 3.2 and it consists of the following components:

A set of links, describing the neuron inputs, with weights $w_0, w_1, \dots, w_m$

A summation function shown in Equation 3.1 for computing the weighted sum u of the inputs x

$$u = \sum_{i=0}^m w_i x_i \quad \text{Equation 3.1}$$

Where u is the weighted sum, w is neuron weight, x is neuron input, and m is the number of inputs

An activation function in Equation 3.2 for limiting the amplitude of the neuron output

$$y = \varphi(u) \quad \text{Equation 3.2}$$

where φ is the activation function and y is the neuron output.

Figure 3.2 A Simple Artificial Neuron

Similar to the biological model, an artificial neural network can learn relationships between a set of inputs and a set of outputs. During the learning process, the artificial neural network extracts useful information from the exemplars (also known as training data) and adjusts its synaptic weights accordingly. It will subsequently be able to classify data on which no prior training was done.

3.2 Network Architectures

The network architecture defines the interconnection of neurons and determines the property of the neural network. In general, there are three fundamentally different classes of the network architectures.

3.2.1 Single-Layer Feedforward Networks

A single-layer feedforward neural network has an output layer of neurons with signals fed from an input layer of source nodes. The input layer of source nodes is not counted because no computation is performed here. Such a network is illustrated Figure 3.3. A typical example of this class of neural network is the single layer Perceptron.

Figure 3.3 Single-Layer Feedforward Network

3.2.2 Multilayer Feedforward Networks

This class of neural network distinguishes itself by the presence of one or more hidden layers, whose nodes are correspondingly called hidden neurons or hidden units. The function of hidden neurons is to intervene between the external input and the network output in some useful manner. By adding one or more hidden neurons, the network is enabled to extract higher-order statistics [32]. In a rather loose sense the network acquires a global perspective despite its local connectivity due to the extra set of synaptic connections and the extra dimension of neuron interactions [27].

The second layer of neurons takes input signals from the input (first) layer of source nodes, feeds its output signals forward to the third layer, and this continues for the rest of the network. The network in Figure 3.4 shows a 3-4-2 network because it has 3 input source nodes, 4 hidden neurons and 2 output neurons. The popular Multilayer Perceptron (MLP) is this type of neural network.

Figure 3.4 Multilayer Feedforward Network

3.2.3 Recurrent Networks

A recurrent neural network distinguishes itself from a feedforward neural network in that it has at least one feedback loop [32]. Figure 3.5 shows a recurrent network with each neuron feeding its output signals back to all the neurons. There is also a hidden neuron in this neural network.

The presence of feedback loops has a profound impact on the learning capability of the network and on its performance. Information about past inputs is fed back into and mixed with the inputs through recurrent or feedback connections for hidden or output units. In this way, the neural network contains a memory of the past inputs via the activations. Moreover, the feedback loops involve the use of particular branches composed of unit-delay elements (denoted by z^{-1}), which result in a nonlinear dynamic behaviour, assuming that the neural network contains nonlinear units [32]. The Hopfield network is a typical example of this class.

Figure 3.5 Recurrent Network with Hidden Neuron

Chapter 4 Genetic Algorithms

4.1 Introduction

Natural evolution is characterized by the survival of the fittest, as described by Charles Darwin. It implies that the fitter individuals survive longer and remain to reproduce fitter offsprings with their genetic information passed on. The idea to apply natural evolution to software is not new. John Holland from University of Michigan defined the concept of genetic algorithms in 1975 and concluded that biology could provide a metaphor for artificial systems [34]. Genetic Algorithms (GAs) are guided by population, yet random search algorithms for complex optimization problems and are based on principles from natural evolutionary theory.

A typical GA process is illustrated in Figure 4.1. To obtain solutions, the problem space is initially encoded into a relevant format, suitable for evolutionary computation. The parameters of the search space are encoded in the form known as *chromosomes* and each indivisible parameter in a chromosome is called a *gene*. A collection of such strings is called a *population*. Initially, a random population is created, which represents different points in the search space. An *objective* and *fitness* function is associated with each string that represents the degree of *goodness* of the chromosome. Based on the principle of survival of the fittest, a few of the strings are reproduced and each is assigned a number of copies that go into the mating pool. Biologically inspired operators like *crossover* and *mutation* are applied on these strings to yield a new generation of chromosomes. The

process of reproduction, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied. [37]

Figure 4.1 The GA process

4.2 GAs for Optimization

4.2.1 Introduction to Optimization

Optimization is a process of attempting to find an optimal solution to the problem under consideration. A large number of problems in various fields can be formulated as optimization problems such as Traveling Salesman Problem (TSP), component placement of Printed Circuit Board (PCB), and digital filter design with the lowest order. A general optimization problem can be defined as the following form:

Minimize objective function: $f(x)$, $x \in \mathfrak{R}^n$

Subject to constraints: $h(x)$

4.2.2 Robustness of Conventional Search Techniques

There are three main types of conventional optimization methods: calculus-based, enumerative, and random [11].

4.2.2.1 Calculus-Based Methods

Calculus-based methods have been heavily studied and subdivided into two main classes: indirect and direct methods [8]. Direct search methods seek local optima by hopping on the objective function and moving in a direction related to the local function gradient. This is a simple concept of hill-climbing. On the other hand, indirect methods search for local extrema by solving the set of equations resulting from setting gradient of the objective function equal to zero. This is the generalization of the elementary calculus concept of extremal point. Although these methods have been advanced, expanded, and modified in many details, they seem to lack the required robustness.

First, they depend on the existence of derivatives which do not exist in many cases. Second, both direct and indirect methods can only achieve local optima. Third, calculus-based methods require objective function which may not exist sometimes. It is clear that these methods depending upon the restrictive requirements of continuity and derivative existence are suitable for very limited problem domains. Therefore, calculus-based methods are often rejected because of their insufficient robustness in unintended domains.

4.2.2.2 Enumerative Methods

Enumerative, search-for-the-best schemes have taken a variety of forms [8]. Their consideration in the robustness race, however, must be limited for lack of efficiency. Most parameter spaces are simply too large to search one at a time and less clever enumerative schemes are rejected for real problems.

4.2.2.3 Random Algorithms

Random algorithms have achieved increasing popularity as researchers recognize the shortcomings of calculus-based and enumerative schemes [8]. Yet, strictly random search must also be discounted because of the efficiency requirement. Random search, in the long run, can be expected to do no better than enumerative schemes.

GAs are computationally simple yet powerful search procedure which uses random choice as a tool to guide highly exploitative search through the parameter space. They do not require the search space to be continuous, differentiable, unimodal or of a functional form. Thus, GAs are suitable for complex optimization problem.

Chapter 5 Hybridization

In the area of data mining, there are many tasks involved in the process of analyzing or identifying a pattern that need appropriate parameter selection and efficient search in complex spaces in order to obtain optimum solutions. Therefore, the application of GAs for solving certain problems of pattern recognition appears to be appropriate and natural [37].

In their purest form, GAs are blind search procedures: they exploit only the coding and the objective function to determine plausible trials in the next generation. This indifference to problem-specific information in large part gives GAs their broad competence. On the other hand, not using all the knowledge available in a particular problem puts GAs at a disadvantage when compared with methods that do use specific problem information. GAs may be combined with various problem-specific search techniques to form hybrid algorithms which exploit the global perspective of the GA and the local convergence of the problem-specific technique.

5.1 *GA-Clustering*

GA-clustering is a genetic algorithm-based K-means clustering technique. The searching capability of GAs has been used for the purpose of appropriately determining a fixed number K of cluster centres in R^N ; thereby suitably clustering the set of n unlabelled points. It is shown in [37] that GA-clustering technique does not suffer from the limitation of the K-means algorithm which is known to provide sub optimal clustering

depending on the choice of the initial clusters and GA-clustering provides significantly superior performance to that of the K-means algorithm [37].

5.2 GA-DistAl

GA-DistAl is a wrapper-based multi-criteria approach to feature subset selection using a genetic algorithm in conjunction with a relatively fast inter-pattern distance-based neural network learning algorithm [18]. The neural network called DistAl in [17] is employed to evaluate the fitness of candidate feature subsets in the genetic algorithm.

DistAl is a simple and fast constructive neural network learning algorithm for pattern classification. The key idea behind DistAl is to add hidden neurons one at a time based on a greedy strategy which ensures that the hidden neuron correctly classifies a maximal subset of training patterns belonging to a single class. Correctly classified examples can then be eliminated from further consideration. The process terminates when this process results in an empty training set [18].

GA-DistAl employs a standard binary coded genetic algorithm with rank-based selection strategy. The binary chromosome of the genetic algorithm represents whether attributes are selected (bits with value 1) or not (bits with value 0). The fitness function combines two criteria: the classification accuracy and the cost of performing classification.

5.3 GA-RBF

The performance of RBF networks is greatly affected by their topology, that is, the choice of centroids making up the hidden layer. A GA is used to evolve fuzzy centroids that become the basis functions of the network's hidden layer. Both the number of centroids per class and their positions are optimised [1].

GA-RBF employs a real-valued coded genetic algorithm. Each chromosome has variant length, meaning different number of centroids, and encodes on set of centroids starting positions for each class. GA-RBF's training set is split into a clustering set and an evaluation set. The objective fitness function used for evaluating individuals consists of the application of the K-means algorithm to the clustering set using the starting positions encoded in the individual's genes, followed by a test classification of the evaluation set using the K-means-computed centroids and the nearest-attracting prototype (NAP) classifier. In addition, a punishment function is used to combat over-fitting and an aging function is used to prevent saturation by super-individuals.

Chapter 6 **K-Means Fast Learning Artificial Neural Network**

Fast Learning Artificial Neural Network (FLANN), first introduced by [9], [20], is a neural network developed with the architecture of Leader Algorithm in [14] enhanced by the vigilance factor from ART. It inherits the positive features of Leader Algorithm [21]:

- Training in a single iteration
- Constraints on individual elements of input vectors (tolerance value)
- Addition of new output nodes upon the formation of new clusters
- Supervised or unsupervised learning
- Architecture can be parallelized

However, a few negative features of Leader Algorithm were also propagated to the FLANN. The main drawback is its inability to centralize its clusters. As a result, clustered exemplars may sometimes switch membership between clusters due to re-arrangement of data. Therefore, a K-means algorithm was introduced into the FLANN algorithm to reshuffle the data in a few passes, so that consistent centroid can be formed regardless of data presentation sequence [22]. The idea of data pattern reshuffling will be discussed later on in Section 6.3.

6.1 *Architecture of KFLANN*

The basic architecture is shown in Figure 6.1. It has 2 layers, the input and output layer, and a set of weight vectors connecting the 2 layers. KFLANN is a fully connected network. The number of output nodes can increase according to the classification

requirements, determined indirectly by the tolerance δ and vigilance ρ parameters. As each new cluster is formed, a new output node is created and the weight vectors of the new output node are assimilated with the exemplar values.

Figure 6.1 Architecture of KFLANN

6.2 Algorithm of KFLANN

Notation	ρ :	vigilance value
	δ_i :	tolerance value of the i^{th} attribute
	n :	the number of input attributes
	I_i :	the i^{th} input node
	W_{ji} :	weight connecting the i^{th} input node and the j^{th} output neuron
	$D[a] = 1$ if $a > 0$. Otherwise $D[a] = 0$.	

- 1 Initialize network with ρ between 0 and 1. Determine δ_i for $i = 1, 2, 3, \dots, n$.

The values of ρ and δ affect the behaviors of the classification and learning process.

- 2 Present the next pattern to the input nodes. If there are no output clusters present, GOTO 6.
- 3 Determine the set of clusters that are possible matches using Equation 6.1. If there are no output clusters GOTO 6.

$$\frac{\sum_{i=1}^n D[\delta_i^2 - (W_{ji} - I_i)^2]}{n} \geq \rho \quad \text{Equation 6.1}$$

- 4 Using criteria in Equation 6.2 determine the winning cluster from the match set from Step 3. Normalize W_{ji} and I_i . The following distance is calculated between the normalized versions.

$$winner = \arg \min_j \left[\sum_{i=0}^n (W_{ji} - I_i)^2 \right] \quad \text{Equation 6.2}$$

- 5 Winner is found. Add vector to the winning cluster. If there are no more patterns, GOTO 7. Else GOTO 2.
- 6 No match found. Create a new output cluster and perform direct mapping from input vector into weight vector of new output cluster. If there are no more patterns, GOTO 7. Else GOTO 2.
- 7 Re-compute cluster center using K-means algorithm. Find the nearest vector to the cluster center in each cluster using Equation 6.2. Place the nearest vector in each cluster to the top of the training data and GOTO 2.

After each cycle of clustering with all exemplars, the cluster centers are updated using K-means algorithm. This is *Step 7* of the KFLANN algorithm. A comparison between each cluster center and patterns in respective cluster is then conducted to determine the nearest point to each cluster center. The algorithm then assigns this point as the new centroid. This method reshuffles the data in order to stabilize the centroid formation [22] and will be discussed in the next section.

6.3 Data Pattern Reshuffling

As mentioned earlier, being derived from the Leader Algorithm [14], the FLANN is also flawed with cluster inconsistencies caused by varying data pattern sequences. The sensitivity to data pattern sequences was addressed by [21], but the solution presented was incomplete. The recommended technique required to perform the entire calculation of cluster centroids for every pattern exemplar presentation. The re-calculation needed at every stage of the algorithm imposed a heavy computational overhead on the network, making it computational intensive for larger data sets. The data pattern reshuffling was later refined in [22] to overcome this limitation.

In human behavioural terms, we reference classes of objects using some form of stereotyping process. The cluster centroids in the KFLANN algorithm can be deemed as a form of stereotyping of a specific class. Other non-centroid exemplars will therefore fall into the categorized spaces defined by the typical centroid prototype. In the mathematical sense, these spaces are analogous to the n-dimensional Voronoi spaces formed by the centroids. It is suggested that the formation of this stereotyping information in daily experiences is linked to the re-organization of recently learnt information which is combined with information already residing within the brain. Such information reorganization is necessary to support the plasticity and stability dilemma, where new information is packed into existing information through a re-organization of stored information. This theory was inspired by the supralevel activations of the brain during sleep periods. Laboratory recordings using EEG and fMRI scans have shown that the

brain enters periodic activations during the Non Rapid Eye Movement (NREM) and Rapid Eye Movement (REM) stages. This is despite the fact that the Thalamus is completely turned off, isolating the brain from all the other sensory percept on the body. During these stages of sleep, the brain seems to enter a hyperactive mode similar to wake periods. The interesting difference in some of these stages, as compared to waking periods, is the lack of external stimulus and the subject is still able to experience active interaction within the dreams. Some evidence shows that laboratory mice indeed replay experiential activities forward and backwards within the brain. Studies at the Wilson Laboratories at MIT indicate that experimental mice undergo extensive replays of experiential events, such as running through a new maze. It was also noted that when the mice entered a sleep process immediately after the new maze experience, the brain activities indicate exact replications of the maze exploration. While the REM and NREM stages are known and published facts found in neuroscience literature, it is still not possible to definitively conclude that these processes are due to consolidating processes within the brain. If this assumption was permitted, then perhaps stereotyping or centroid stability can be achieved by some form of information replay within the designed algorithm of FLANN.

The KFLANN algorithm extends this reasoning from the FLANN and introduces data reshuffling into the neural network learning process. While this is a small step of repetition, the results indicate that this process was sufficient to stabilize the centroids (stereotypes) and thus created consistent clusters. The reshuffling process involves the

movement of data points closest to the centroids to top of the data list. This process changes the seeding sequence on the data presentation sequence. The entire clustering process is then repeated on the new reshuffled data list. When no further changes can be observed, the algorithm is said to have converged and the centroid stabilization achieved. This data reshuffling process is introduced as step 7 in KFLANN algorithm shown in the previous section.

6.4 Network Parameter Settings

The KFLANN algorithm is able to cluster effectively only if the correct δ and ρ values are used [23]. Both parameters indirectly determine the cluster assignments and behaviour of the neural network. If there is no knowledge of cluster assignments or the number of clusters at all, it is a great challenge to determine the appropriate tolerance and vigilance value [21].

6.4.1 Tolerance Value

Tolerance value (δ) is the maximum distance of how far a pattern in a cluster can be from the cluster center in each attribute. It is called threshold in Leader Algorithm. As illustrated in Figure 6.2, the δ values indirectly determine the clustering behaviour of the algorithm. A larger δ provides lesser clusters (a), while a smaller δ provides more clusters (b).

KFLANN suffers the same difficulty in determining tolerance as the threshold in Leader Algorithm. Since the search space of tolerance is continuous, nonlinear and multimodal,

it is very difficult to locate the global optimal. Moreover, there is no exact functional form to specify the tolerance setting, which makes it very hard to apply the conventional optimization methods. Since the characteristic spread of data is also sometimes unknown, the δ values are still very much a guessing game.

Figure 6.2 Different clusters are formed for the same data set when δ is varied

6.4.1.1 Domain Knowledge

The presence of domain knowledge is perhaps the most helpful form of tolerance setting. This approach is able to produce an acceptable clustering result since the domain knowledge is usually superior [22]. However, domain knowledge is usually not available especially for large data sets of unknown significance.

6.4.1.2 Standard Deviation of the Attributes

In the field of statistics, most data distributions in the natural world tend to fall into a typical bell-shaped curve which is also known as typical Gaussian distribution or normal distribution. Figure 6.3 shows the normal distribution curve and the coverage of the entire data range under different standard deviations, where μ refers to the mean and σ refers to

the standard deviation. According to the Empirical Rule [7], the normal distribution curve has the following features:

- About 68% of the data will fall in the interval between $\mu - \sigma$ to $\mu + \sigma$.
- About 95% of the data will fall in the interval between $\mu - 2\sigma$ to $\mu + 2\sigma$.
- About 99.7% of the data will fall in the interval between $\mu - 3\sigma$ to $\mu + 3\sigma$.

Figure 6.3 Gaussian Distribution Curve

When the data is not of normal distribution, the Empirical Rule is not applicable. The reference would then require the Chebyshev's Rule [7], which indicates that

- About 75% of the data will fall in the interval between $\mu - \sigma$ to $\mu + \sigma$.
- About 89% of the data will fall in the interval between $\mu - 2\sigma$ to $\mu + 2\sigma$.
- About 97% of the data will fall in the interval between $\mu - 3\sigma$ to $\mu + 3\sigma$.

Therefore, it is suggested that at least 6 standard deviations (3σ on each side of μ) of the mean will approximately cover the entire data space for both normal and non-normal

distribution domain. With this, the standard deviation of the feature space can be used to initialize the feature tolerance and facilitate a heuristic means for the KFLANN to form relevant clusters and correctly represent class distributions.

The standard deviation summarizes the dispersion of individual values from the mean. This would be a reasonable yet crude tolerance value measure. And this method is susceptible to outlier values, which exist in most data sets. The tolerance (δ) using standard deviation (σ) is show in Equation 6.3.

$$\delta_i = \sigma_i = \sqrt{\frac{\sum_{j=1}^n (x_{ji} - \mu_i)^2}{n}} \quad \text{Equation 6.3}$$

Where x_{ji} is the i^{th} feature of the j^{th} pattern and μ is the respective mean value

6.4.1.3 Binary Search Algorithm

Another approach is gradual determination of tolerance values based on the maximum and minimum differences in the attribute values. The algorithm resembles a binary search and is presented as follows [21]:

Initialize $\delta_i = (\partial_{imin} + \partial_{imax})/2$

while number of clusters in not appropriate

 run standard KFLANN based on current δ_i values.

 if number of clusters is less than expected

$$\delta_i = (\delta_i + \partial_{imin})/2$$

 else if number of clusters is more than expected

$$\delta_i = (\delta_i + \partial_{imax})/2$$

end while

δ_i - Tolerance value for attribute i .

∂_{imin} - Minimum difference in attribute values for attribute i . This is the smallest difference between values of the attribute.

∂_{imax} - Maximum difference in attribute values for attribute i . This is simply the difference between the smallest and largest values of the attribute

However, it is hard to determine which tolerance values to modify and thus, it is basically a trial and error procedure. Moreover, the domain knowledge of data is required to decide whether the number of clusters formed is appropriate.

6.4.1.4 Exhaustive Search Algorithm

This algorithm is a brute force combinatorial exhaustive search, which tries all possible combinations of tolerance and vigilance values. This is appropriate because there is no assumed knowledge of the data and it removes the requirement of knowing the data or desired class assignments [40]. However, being an exhaustive search methodology, it is inevitable that the algorithm is slow.

The method essentially combines the different tolerance values in step-wise intervals. In an exemplar with m attributes and n step-wise intervals n^m modifications have to be made on the tolerance values. This means that as the number of attributes grows linearly, the time taken to complete the exhaustive search grows exponentially. Moreover, it is difficult to determine the correct resolutions of tolerance values. Small resolutions will

result in polynomial growth of computation while big resolutions may miss good tolerance values.

6.4.2 Vigilance Value

Vigilance value (ρ) is the number of features out of the total that should be used for similarity measurement [22]. For example, if a pattern consists of 5 input attributes and 4 similar features are required for clustering, ρ is 0.8.

$$\rho = \frac{f}{n} \quad \text{Equation 6.4}$$

Where f is the number of features required to be classified in a same cluster and n is the total number of features.

Time taken to search for correct vigilance value grows linearly with respect to the linear increase of the number of input attributes according to its formula. For example, a data set has vectors with 5 attributes. Since the value of vigilance is defined as $\rho = f/5$, $f = 1, 2, 3, 4, 5$, it can only have values from the set $\{0, 1/5, 2/5, 3/5, 4/5, 1\}$ [21]. So it is actually a discrete parameter (with floating point values). The best way to change the vigilance value is to update it by $1/n$.

6.5 Feature Selection

Although input data was preprocessed, there may still be features that are irrelevant or not of interest in the data set. Feature selection can help identify the key features, reduce

the dimension of the data set, and thereby improve the clustering accuracy of KFLANN. In the previous research project, which will be mentioned in Section 6.7, the feature selection of KFLANN made use of an exhaustive search algorithm in order to locate the optimal subset of features. However, this approach further degraded the clustering speed of KFLANN in data mining.

6.6 KFLANN using Standard Deviation (SD-KFLANN)

SD-KFLANN makes use of the standard deviation of feature space as its tolerance values to form relevant clusters and correctly represent class distribution [19]. Four data sets, Iris, wine, Wisconsin Breast Cancer (WBC), and glass, were used in an experiment to test the performance of SD-KFLANN. Interested readers can refer to Appendix A.1, A.2, A.3 and A.4 respectively for more details of the four data sets. Table 6.1 shows the experimental results and also compares the results with the ones obtained by K-NN. SD-KFLANN produced lower clustering accuracy on all the other data sets except Iris.

Table 6.1 Experimental Results of SD-KFLANN on the Three Data Sets

Data Set	SD-KFLANN	K-NN
Iris	100%	95.1%
Wine	90.77%	96.7%
WBC	91.77%	96.6%
Glass	69.05%	72%

The above comparison shows that standard deviation is a reasonable yet crude tolerance value measure. More suitable tolerance settings were still to be investigated.

6.7 KFLANN using Exhaustive Search (ES-KFLANN)

ES-KFLANN employs exhaustive search algorithm in [40] in order to make a thorough search on the network parameter space and obtain significant data features as well as accurate clustering results. Three data sets, Iris, wine, and Wisconsin Breast Cancer (WBC), were used in the experiment to test whether the algorithm could indeed achieve the above objectives. Interested readers can refer to Appendix A.1, A.2, and A.3 respectively for more details of the three data sets. The experimental results are shown in Table 6.2.

Table 6.2 Experimental Results of ES-KFLANN on the Three Data Sets

Data Set	Dimension	Accuracy	Time
Iris	1	98%	5 minutes
Wine	3	95.51%	> 2 weeks
WBC	5	96%	> 2 weeks

Table 6.3 Comparison of Clustering Accuracy Between ES-KFLANN and K-NN

	Iris	Wine	WBC
ES-KFLANN	98%	95.51%	96%
K-NN	95.1%	96.7%	96.6%

Comparison with K-NN is given in Table 6.3, the KFLANN with exhaustive search algorithm achieved very good results. It was also able to retrieve significant factors among all the features in the data sets. However, the time taken to complete clustering

was too long for the algorithm to be practical. A less expensive search method for the network parameters has to be established to make the KFLANN practical and efficient.

As previously mentioned, the search space of tolerance of KFLANN is continuous, nonlinear, multimodal, and of no functional form. The curse of dimensionality is the main problem encountered in efficient search of proper tolerance setting. The ability to search through the complex parameter space and likeliness to reach a near-optimal condition regardless of dimensionality is the strong motivation for introducing GA into KFLANN.

Chapter 7 Hybrid Model of GA-KFLANN

The architecture of the GA-KFLANN is illustrated in Figure 7.1. The original KFLANN takes tolerance and vigilance values produced by the GA to cluster the input data set with selected features by GA. After the KFLANN converges or a predefined number of cycles have been reached, the fitness values of the clustering results are evaluated and fed back to GA to generate the next population of better parameters. This process continues until a preset number of generations of GA have been reached or no much improvement on the fitness value can be observed.

Figure 7.1 The Architecture of GA-KFLANN

7.1 Chromosome Encoding

The chromosomal encoding for the GA-KFLANN consists of two parts: the control genes (ConG) and the coefficient genes (CoeG). The ConG are a string of binary numbers, which are used to turn on or off the corresponding features to achieve the goal of feature selection. Whereas, the CoeG are a sequence of real numbers representing tolerance and

vigilance values to control the behaviour of KFLANN. The ConG may not be used when all features are fully utilized in a clustering problem. For the purpose of discussion, the Iris flower dataset is now used as an example to illustrate the encoding required for GA computations. The δ and ρ are first converted into chromosomal equivalents as shown in Figure 7.2. CoeG shown in shaded pattern represent those turned off by their corresponding ConG.

Population									
Control Genes (ConG)					Coefficient Genes (CoeG)				
					Tolerance δ				Vigilance ρ
					Sepal length	Sepal width	Petal length	Petal width	
1	1	1	1	0	0.88	0.56	0.40	0.59	0.75
2	0	0	1	1	0.54	0.69	0.17	0.47	0.5
$\vdots$									
n	1	1	1	1	0.73	0.65	0.34	0.12	1.0

Figure 7.2 Sample Chromosome Encoding for the Iris Dataset

7.2 Population Initialization

In the 1st generation of the GA, ConG (if used) are randomly assigned to the value ‘1’ or ‘0’, while CoeG are randomly initialized to values between the upper bound and the lower bound of tolerance or vigilance of features from the input data. Since tolerance value (δ) is the maximum distance of how far a pattern in a cluster can be from the cluster center in each attribute [39] as shown in Figure 6.2, the upper bound of tolerance for each attribute can be set to half of the maximum distance among data points in that attribute, while the lower bound can be assigned to the minimum distance. For example, assume

there are 5 data points: (0, 0), (1, 1), (3, 4), (5, 0), and (2, 6). The upper bound for the 1st dimension is $(5-0)/2 = 2.5$, while the lower bound is 1, which is the minimum distance among data points. Therefore, tolerance δ_1 shall be initialized in the range [1, 2.5]. Similarly tolerance δ_2 can be found in the range [1, 3].

7.3 Fitness Evaluation

Fitness evaluation is the key issue for GA. A good fitness evaluation can help the GA produce proper tolerance and vigilance values, leading the KFLANN to the optimal clustering, while a poor one can cause GA to converge towards a wrong direction and lead to inaccurate clustering. Fitness evaluation is straightforward for supervised learning. The classification accuracy can be used as the fitness function in search for optimal network parameters. However, it is hard to find a proper fitness function for unsupervised learning.

Within-group variance $\sigma_{W_i}^2$ for each cluster i and between-group variance σ_B^2 can be easily computed from the output of KFLANN for each clustering result. And the two types of variance satisfy the following equation:

$$\sigma_T^2 = \sigma_W^2 + \sigma_B^2 \quad \sigma_W^2 = \sum_{i=1}^k \sigma_{W_i}^2 \quad \text{Equation 7.1}$$

where σ_T^2 is the total variance. Since σ_T^2 is fixed for a data set, a natural criterion for grouping is to minimize σ_W^2 , or, equivalently, maximize σ_B^2 [1].

Moreover, the clustering with the maximum between-group variance and minimum within-group variance means highly dense clusters and good data compression. Thus, a possible evaluation criterion can be formed as maximizing the term: σ_B^2 / σ_W^2 .

It works reasonably well for data sets without overlapping patterns, but not so well as expected with overlapping clusters. An additional term used in fitness evaluation is a Boolean variable, convergence, representing whether a clustering converges. The whole term is expressed as follows:

$$fitness = (convergence + 1) \times \sigma_B^2 / \sigma_W^2 \quad \text{Equation 7.2}$$

This is to ensure that converged clustering has much higher fitness value and force the GA to produce tolerance and vigilance that can make KFLANN converge and form consistent centroids.

7.4 Reproduction

Solution strings from the current generation are copied into a mating pool according to the corresponding fitness values. Strings with higher fitness values will likely be represented in higher numbers in the mating pool, which means δ and ρ generating higher classification/clustering accuracy will more likely survive and pass their values to the next generation. This is because that there is a higher chance for δ and ρ with higher classification/clustering accuracy to hit the respective correct settings.

Stochastic Universal Sampling (SUS) is the most popular reproduction strategy and utilized in this research project. SUS is basically a roulette wheel which rotated once with n evenly spaced markers, where n is the population size. Each chromosome is assigned a portion of the wheel's perimeter in proportion to its fitness value. The n markers will choose n chromosomes into the mating pool. For example, a population of 4 chromosomes with different fitness values is shown in Table 7.1.

Table 7.1 Sample Chromosomes and Fitness values

Number	Chromosome	Fitness	% of Total	No. selected
1	01101	13	13	1
2	01111	30	30	1
3	10101	8	8	0
4	11111	49	49	2
Total		100	100	4

The 4 chromosomes are assigned different portions of the wheel based on their respective fitness values as shown in Figure 7.3. In SUS, it is clear that a chromosome with higher fitness is more likely to have more copies in the mating pool, but it also prevents the fitter individuals from dominating the whole population.

Figure 7.3 Roulette Wheel of SUS

7.5 Crossover

Crossover is a probabilistic process that exchanges information between two parent chromosomes in the mating pool for generating two child chromosomes, so that proper settings of parameters can be grouped together into a single child chromosome. For example, one parent has the best setting of sepal length, while another has proper petal width value. A better clustering result will be achieved if the 2 good settings can be grouped together into just one offspring.

Two types of crossover operators are implemented in this research since the ConG and CoeG make use of different encoding schemes. Uniform crossover is applied to the ConG while convex crossover is applied to the CoeG. In uniform crossover a template of the same length as ConG is randomly generated to decide which parent to contribute for each bit position. For example, 2 parents are shown in Table 7.2, one is underlined and the other is italic. Bits from parent 1 are passed to offspring 1 if the corresponding bits in the

template are of value '1'; otherwise those bits are passed to offspring 2. This rule works in reverse way for parent 2. Therefore, the 1st four and last two bits of parent 1 are passed to offspring 1, while the rest goes to offspring 2. Similarly, parent 2 contributes different parts to offspring 1 and 2 respectively according to the template.

Table 7.2 An Example of Uniform Crossover

	Parent	Template	Offspring
1	1001011	1111001	1001 <u>1</u> 01
2	0101101		0101 <u>0</u> 11

If x and y are the CoeG of two parents, then convex crossover is of the following form:

$$x' = \lambda x + (1 - \lambda)y$$

Equation 7.3

$$y' = (1 - \lambda)x + \lambda y$$

Equation 7.4

where x' and y' are the corresponding CoeG of two children and $0 < \lambda < 1$. λ is set to 0.7 in this research project. Convex crossover is the one most commonly used and has the advantage of maintaining feasibility when the feasible set is convex.

7.6 Mutation

Mutation operation randomly picks up a gene in the generated offspring strings and changes its value properly in order to allow the GA to escape from a local optimal to search for a global optimal. Two types of mutation operators are used in this research like the described crossover above. Normal mutation is applied to the ConG while dynamic mutation is applied to the CoeG.

For a given string, if the gene x is selected for mutation, then the offspring $x' = 1 - x$ if x is a control gene. If x is a coefficient gene, then x' is selected with equal probability from the two choices:

$$x' = x + r(u - x)(1 - \frac{t}{T})^b \quad x' = x - r(x - l)(1 - \frac{t}{T})^b \quad \text{Equation 7.5}$$

$$x \in [l, u]$$

r : a random number chosen uniformly from $[0, 1]$

t : current generation number

T : the maximum number of generations

b : degree of nonuniformity.

The strength of non-uniform mutation decreases as the generation advances, so that the search gradually turns from initial global search to local tuning of solutions. The case $b = 0$ reduces to the standard, uniform mutation of replacing a number with one chosen randomly from the interval $[l, u]$. Detailed study of non-uniform mutation can be found in [42].

7.7 Population Replacement

In a simple GA, one entire generation of population is replaced by the offspring and this is called non-overlapping generations. However, it is possible that offsprings become weaker than the parents as some good genes in the parents may be lost. And evolution based on weak generations may not be able to reach global optimum. To overcome this

problem, a portion of the previous generation is retained in the new population, which is called overlapping generations.

$(\mu + \lambda)$ selection is used in this research in order to maintain a steady performance throughout the evolution of the GA and achieve global optimum, where μ best parents and λ best offspring are selected as the population of the next generation. As the best solutions are always passed to the next generation in $(\mu + \lambda)$ selection, the population will become stronger than or at least the same as the previous generation. Another reason to choose this population replacement strategy is because that μ and λ can be adjusted during the evolution to enable the use of a local search method, which will be introduced in the next section, to further improve the search effectiveness.

7.8 Crossover Hill-Climbing (XHC)

Hill-climbing is a local search algorithm that commences from a single solution point. At each step, a candidate solution is generated using a move operator of some sort. The algorithm simply moves the search from the current solution to a candidate solution if the candidate has better or equal fitness. Crossover hill-climbing (XHC) was first described by [35] and [38] as a special crossover local search approach. Its basic idea is to use hill-climbing as the move accepting criterion of the search and use crossover as the move operator.[24]

In this research project, GA-KFLANN started with $\mu = 2$ and $\lambda = 18$, meaning 2 best parents and 18 best offspring were selected into the next generation. As the generation advanced, the ratio of parents to offspring in the next generation increased from 1:9 to 9:1, which introduced less and less variations in generations and the population became more and more similar. This gradually lowered the search diversity and the crossovers created offspring distributed densely around the parents, favouring local tuning [24]. Although the same crossover operators are applied, the search is initially started at global space and limited in fit areas located by GA and local tuning is improved. It is more likely to reach global optimal by locating the global fitter zone first followed by local searching using this way.

Chapter 8 Experiments and Results

Five real-world datasets obtained from the UCI Machine Learning Repository [5] were used to explore the feasibility of GA-KFLANN. In order to illustrate the improved performance of GA-KFLANN, the clustering results of other three methods were included in this section for comparison. These are the KFLANN using Exhaustive Search (ES-KFLANN), KFLANN using Standard Deviation (SD-KFLANN) and the K-Nearest Neighbour (K-NN). Of the three methods, ES-KFLANN was original contribution, whereas SD-KFLANN and K-NN can be found in [19] and [30] respectively.

8.1 *Iris Data Set*

Fisher's paper [28] is a classic in the field and is referenced frequently to this day. Interested readers please refer to Appendix A.1 for more details of Iris data set. The data set contains 150 random samples of flowers from the Iris species: Setosa, Versicolor, and Virginica. From each species there are 50 observations with 4 attributes measured in centimetres.

A population size of 20 was used and the algorithm was run for 40 generations. Each chromosome in the population consisted of 4 ConG for feature selection and 5 CoeG for four δ values and one ρ value, as shown in Figure 7.2. The 4 ConG was randomly initialized to the value '1' or '0' to select or deselect the corresponding attributes of Iris data. Each CoeG was randomly initialized in the value range mentioned in the last chapter to represent the respective δ and ρ values. A few results were obtained by tuning

the settings of the GA. Comparisons were made to determine the best settings for the GA. Results shown in the next sections highlight the plots of Accuracy against the Generation number. The results shown are the best of three runs of each setting.

8.1.1 Performance of GA-KFLANN with Non-Overlapping and Overlapping Generations

Figure 8.1 shows the maximum accuracy obtained in each generation which was used to illustrate the performance of both settings. Two methods, namely non-overlapping and overlapping generations were investigated here. The technique with overlapping generations used here is $(\mu + \lambda)$ selection introduced in the last chapter.

Figure 8.1 Performance of GA-KFLANN with Non-Overlapping and Overlapping Generations

Results indicate that the GA-KFLANN with non-overlapping generations produced peaks and valleys in the plot. Taking a portion of the figure for example, the maximum accuracy of 0.83 (point A), 0.94 (point B), and 0.87 (point C) were obtained from generation 6 to 9. The portion of the plot from A to B is good because the maximum

accuracy is increasing with the generations, while the part from B to C is falling because the maximum accuracy is decreasing, which means that the offspring is weaker than the parents. This shows that the peaks and valleys in the results are unstable and the offspring can be worse than the parents during the evolution. And in the last generation, results show that the GA-KFLANN with non-overlapping generations failed to achieve global optimum. On the contrary, the plot with $(\mu + \lambda)$ selection shows a stably rising curve and the global optimum, 0.98 accuracy, is achieved in the last few generations. These findings support the theory on non-overlapping generations mentioned in the last chapter. $(\mu + \lambda)$ selection ensures that the good genes from the parents are inherited and offsprings are always better than or at worst, the same as the parents. Therefore, $(\mu + \lambda)$ selection will be employed in all the later experiments.

8.1.2 Performance of GA-KFLANN with and without XHC

Crossover hill-climbing (XHC) was used to perform a diverse search (global search) in the early generations and narrow down the search space to the fit zone (local search) in the later generations. It is more likely to reach global optimal using XHC. The best results obtained in three runs of GA-KFLANN with and without XHC were plotted in Figure 8.2 and Figure 8.3. The reason to plot the average accuracy with and without XHC is because that the maximum accuracy does not provide information for the entire population in each generation, while the average accuracy is the average of the entire population in each generation.

Figure 8.2 Plot of Max Accuracy vs. Generation with and without XHC

Figure 8.3 Plot of Average Accuracy vs. Generation with and without XHC

From Figure 8.2, the results of maximum accuracy obtain by GA-KFLANN with and without XHC are similar. Both achieved 98% accuracy in early generations. However, the results of average accuracy are quite different as shown in Figure 8.3. In the early generations, similar results are viewed because both techniques use diverse global search and the population is randomly distributed over the parameter space. However, the plot of

results with XHC shows higher average accuracy than the one without XHC in the later generations. This is because XHC narrows down the search space/population to the global fitter zone in the later generations and the average accuracy of the fitter zone is supposed to be higher. GA-KFLANN without XHC is still doing diverse global search by then and the average accuracy of the population is lower. The above discussion supports the theory on XHC in the previous chapter. Although it does not guarantee to obtain optimal solutions, XHC provides higher possibility. Therefore, XHC will be employed in the later experiments.

8.1.3 Performance of GA-KFLANN with Uniform Mutation and Dynamic Mutation

The mutation used in CoeG can be uniform mutation ($b = 0$) or dynamic mutation ($b = 1$) as shown in Equation 7.5. The best results obtained in three runs of GA-KFLANN with both mutation methods were plotted in Figure 8.4 and Figure 8.5. The reason to show the plot of average accuracy is because average accuracy provides information for the entire population in each generation, like the last section.

Figure 8.4 Plot of Max Accuracy vs. Generation with Different Mutation Methods

Figure 8.5 Plot of Average Accuracy vs. Generation with Different Mutation Methods

From Figure 8.4, the plot of results obtained using dynamic mutation shows that the maximum accuracy is low in the early generations but high in the later generations. And the plot in Figure 8.5 shows that GA-KFLANN produced results with higher average accuracy using dynamic mutation in the later generations. The possible reason is that dynamic mutation allows the GA to start with a higher amount of mutation so as to search

the whole variable space and lower the amount of mutation gradually to make the final solution settle down at global optimal. This makes a thorough search in the early generations and locates the population to the fitter zone in the later generations. Like XHC, dynamic mutation does not guarantee optimal solutions, but it provides higher possibility to reach global optimal. Therefore, dynamic mutation will also be employed in the later experiments.

8.1.4 Performance of GA-KFLANN with Different Mutation Probabilities (P_m)

As mentioned early, the desired objective of using the GA in KFLANN is to perform a thorough search in the early generations to locate the global fitter zone and locate the optimal solution in this zone in the later generations. Different mutation probabilities were used to find the best setting for this objective. Figure 8.6 and Figure 8.7 show the results of GA-KFLANN with different mutation probabilities (P_m).

Figure 8.6 Plots of Average Accuracy Obtained Using Different Mutation Probabilities

Figure 8.7 Plots of Max Accuracy Obtained Using Different Mutation Probabilities

The maximum accuracy plots in Figure 8.7 look similar and high maximum accuracy is obtained in all. However, in Figure 8.6, the average plot with $P_m = 0.05$ reveals that the convergence begins from very early generations, while the plot with $P_m = 0.4$ converges in very late generations. This means that $P_m = 0.05$ made GA-KFLANN prematurely converge and the parameter space was not thoroughly explored, while the latter one makes GA-KFLANN converge too slowly and the fitter zone was not given enough attention. The plots with $P_m = 0.2$ is chosen as the proper setting for clustering using GA-KFLANN in Iris data.

8.1.5 Performance of GA-KFLANN with Different Crossover Probabilities (P_c)

Different crossover probabilities were used to find the best setting for GA-KFLANN.

Figure 8.8 and Figure 8.9 show the results of GA-KFLANN with different crossover probabilities (P_c).

Figure 8.8 Plots of Average Accuracy Obtained Using Different Crossover Probabilities

Figure 8.9 Plots of Max Accuracy Obtained Using Different Crossover Probabilities

The results obtained here are similar to those discussed in the previous section. With the different crossover probabilities, the plots of results of maximum accuracy look similar and little information can be extracted. A peek into the average accuracy results provides more clues to the intermediate operations.

From Figure 8.8, it can be observed that the accuracy plateaus slower as the crossover probability goes higher. This could possibly mean that higher crossover probabilities introduce more variation into the population of the GA. GA-KFLANN with $P_c = 0.75$ produced an ideal plot, which performed enough diverge search in the early generations and local search in the later generation. Therefore, $P_c = 0.75$ is chosen as the proper setting for clustering using GA-KFLANN in Iris data.

From all the above testing, it is determined that the proper settings for GA-KFLANN on Iris data will be $(\mu + \lambda)$ selection, XHC, dynamic mutation with $P_m = 0.2$, and $P_c = 0.75$.

8.1.6 Comparison with Other Methods

The clustering results and comparisons with other algorithms are shown in Table 8.1. It is clear that all the four clustering techniques achieved high accuracy. Both GA-KFLANN and ES-KFLANN can also extract the significant factor from the four attributes of Iris data. However, GA-KFLANN did all above within 20 seconds rather than 5 minutes for ES-KFLANN from Table 6.2.

Table 8.1 Comparison of Clustering Results on Iris Data

	Dimension	Accuracy
GA-KFLANN	1	98%
ES-KFLANN	1	98%
SD-KFLANN	Full Set	100%
K-NN	Full Set	95.1%

8.2 Wine Data

This data were obtained from a chemical analysis of wines grown within the region of Italy, but were derived from three different cultivators. The analysis determined the quantities of 13 constituents found in each of the three types of wines. There were 178 instances of wine samples in the data set. Interested readers please refer to Appendix A.2 for more details of wine data set.

A population size of 20 was used and the algorithm was run for 40 generations. The GA also made use of $(\mu + \lambda)$ selection, XHC, and dynamic mutation. Because there are 13 attributes in the wine data, one ConG will be required to select or deselect each of them and there will be one tolerance value assigned to each attribute. Therefore, each chromosome in the population consisted of 13 ConG for feature selection and 14 CoeG for thirteen δ values and one ρ value. A few results were obtained by tuning the crossover and mutation probabilities of the GA. Comparisons were made to determine the best

settings for the GA. All the plots afterwards are based on Accuracy vs. Generation from the best result of three runs for each setting.

8.2.1 Performance of GA-KFLANN with Different Mutation Probabilities (P_m)

Different mutation probabilities were used to find the best setting for GA-KFLANN.

Figure 8.10 shows the results of GA-KFLANN with different mutation probabilities.

Figure 8.10 Plots of Average Accuracy Obtained Using Different Mutation Probabilities

The difference of plots in Figure 8.10 is not so obvious comparing what is obtained in Iris testing in Figure 8.6, but still observable. The plot with $P_m = 0.05$ converges early from generation 26, while the plot with $P_m = 0.5$ converges late from generation 34. The plot with $P_m = 0.2$, converging from generation 30, gave more time for initial global search as well as local hill-climbing later on. Therefore, $P_m = 0.2$ is chosen as the proper setting for clustering wine data using GA-KFLANN.

8.2.2 Performance of GA-KFLANN with Different Crossover Probabilities (Pc)

Different crossover probabilities were used to find the best setting for GA-KFLANN.

Figure 8.11 shows the results of GA-KFLANN with different crossover probabilities.

Figure 8.11 Plots of Average Accuracy Obtained Using Different Crossover Probabilities

From Figure 8.11, the plot with $P_c = 0.35$ shows premature convergence of the population of GA-KFLANN, while the plot with $P_c = 0.95$ shows the lack of local search in the later generations. The plot with $P_c = 0.75$ gives better global and local search than both of them. Therefore, $P_c = 0.75$ is chosen as the proper setting for clustering using GA-KFLANN in wine data.

From the above testing, it is determined that the proper settings for GA-KFLANN on wine data will be $(\mu + \lambda)$ selection, XHC, dynamic mutation with $P_m = 0.2$, and $P_c = 0.75$.

8.2.3 Comparison with Other Methods

The result of ES-KFLANN shown in Table 8.2 is not final and the clustering process was not completed because it took too much time. The best result of wine data obtained by ES-KFLANN is used in the following comparison. From Table 8.2, all the four clustering methods achieved high accuracy on wine data. GA-KFLANN obtained a bit more accurate result with a few more dimensions comparing with ES-KFLANN. And GA-KFLANN spent only 1 min 3 sec on clustering the wine data rather than 2 weeks for partial completion of ES-KFLANN as shown in Table 6.2.

Table 8.2 Comparison of Clustering Results on Wine Data

	Dimension	Accuracy
GA-KFLANN	6	96%
ES-KFLANN	3	95.51%
SD-KFLANN	Full Set	90.77%
K-NN	Full Set	96.7%

8.3 Wisconsin Breast Cancer (WBC) Data

This breast cancer data set was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg [39]. There were 699 patterns with 9 attributes in this data set. Interested readers please refer to Appendix A.3 for more details of Wisconsin breast cancer data set.

A population size of 20 was used and the algorithm was run for 40 generations. Each chromosome in the population consisted of 9 ConG and 10 CoeG. The 9 ConG was used to perform feature selection on the 9 attributes of WBC data. Nine out of the ten CoeG represented the nine δ values, each for one attribute of WBC data, and one CoeG represented the ρ value. The GA also made use of $(\mu + \lambda)$ selection, XHC, and dynamic mutation. A few results were obtained by tuning the crossover and mutation probabilities of the GA.

8.3.1 Performance of GA-KFLANN with Different Mutation Probabilities

Different mutation probabilities were used to find the best setting for GA-KFLANN.

Figure 8.12 shows the results of GA-KFLANN with different mutation probabilities.

Figure 8.12 Plots of Average Accuracy Obtained Using Different Mutation Probabilities

Slight difference can be observed from the plots in Figure 8.12. The plot with $P_m = 0.05$ converges early from generation 16, while the plot with $P_m = 0.5$ converges late from

generation 30. The other plot with $P_m = 0.15$, converging from generation 25 gives more time for initial global search as well as local hill-climbing later on. $P_m = 0.15$ is chosen as the proper setting for clustering WBC data using GA-KFLANN.

8.3.2 Performance of GA-KFLANN with Different Crossover Probabilities

Different crossover probabilities were used to find the best setting for GA-KFLANN.

Figure 8.13 shows the results of GA-KFLANN with different crossover probabilities.

Figure 8.13 Plots of Average Accuracy Obtained Using Different Crossover Probabilities

From Figure 8.13, the plot with $P_c = 0.35$ shows slightly faster convergence of the population of GA-KFLANN, while the plot with $P_c = 0.95$ shows the lack of local search in the later generations. The plot with $P_c = 0.75$ gives better results than both of them. Therefore, $P_c = 0.75$ is chosen as the proper setting for clustering using GA-KFLANN in WBC data.

From the above testing, it is determined that the proper settings for GA-KFLANN on WBC data will be $(\mu + \lambda)$ selection, XHC, dynamic mutation with $P_m = 0.15$, and $P_c = 0.75$.

8.3.3 Comparison with Other Methods

The clustering of ES-KFLANN on WBC data was also not completed because it took too much time. The result of ES-KFLANN clustering on WBC data is taken from the best obtained. As shown in Table 8.3, it is clear that GA-KFLANN did very well in the aspects of clustering accuracy and dimension reduction comparing with ES-KFLANN, SD-KFLANN and K-NN. Moreover, it took 5 min 36 sec for clustering WBC data rather than 2 weeks for partial completion of ES-KFLANN from Table 6.2.

Table 8.3 Comparison of Clustering Results on Wisconsin Breast Cancer Data

	Dimension	Accuracy
GA-KFLANN	4	96%
ES-KFLANN	5	96.5%
SD-KFLANN	Full Set	91.77%
K-NN	Full Set	96.6%

8.4 Glass Data

Glass data set comes from forensic testing of glass collected by B. German on 214 fragments of glass. It contains six types of glass with nine attributes. Interested readers please refer to Appendix A.4 for more details of glass data set.

A population size of 20 was used and the algorithm was run for 40 generations. The chromosome setting of glass data is the same as the one of WBC data, because both of them have nine attributes. Each chromosome in the population consisted of 9 ConG for feature selection and 10 CoeG for nine δ values and one ρ value. The GA also made use of $(\mu + \lambda)$ selection, XHC, and dynamic mutation. A few results were obtained by tuning the crossover and mutation probabilities of the GA.

8.4.1 Performance of GA-KFLANN with Different Mutation Probabilities

Different mutation probabilities were used to find the best setting for GA-KFLANN. Figure 8.14 shows the results of GA-KFLANN with different mutation probabilities.

Figure 8.14 Plots of Average Accuracy Obtained Using Different Mutation Probabilities

From Figure 8.14, it is clear that GA-KFLANN with $P_m = 0.1$ and 0.2 converged in very late generations. The other plot with $P_m = 0.05$ both gave more time for initial global search as well as local hill-climbing later on. $P_m = 0.05$ is chosen as the proper setting for clustering glass data using GA-KFLANN.

8.4.2 Performance of GA-KFLANN with Different Crossover Probabilities

Different crossover probabilities were used to find the best setting for GA-KFLANN. Figure 8.15 shows the results of GA-KFLANN with different crossover probabilities.

Figure 8.15 Plots of Average Accuracy Obtained Using Different Crossover Probabilities

From Figure 8.15, the plot with $P_c = 0.65$ shows the lack of local search in the later generations. The plots with $P_c = 0.05$ and 0.25 look similar and both gave enough generations for global and local search. Therefore, $P_c = 0.25$ is chosen as the proper setting for clustering using GA-KFLANN in glass data.

From the above testing, it is determined that the proper settings for GA-KFLANN on glass data will be $(\mu + \lambda)$ selection, XHC, dynamic mutation with $P_m = 0.05$, and $P_c = 0.25$.

8.4.3 Comparison with Other Methods

Table 8.4 shows the comparison of clustering results from three methods. GA-KFLANN obtained the highest accuracy and it was able to select only three features out of nine in the data set.

Table 8.4 Comparison of Clustering Results on Glass Data

	Dimension	Accuracy
GA-KFLANN	3	77%
SD-KFLANN	Full Set	69.05%
K-NN	Full Set	72%

8.5 Pima Indians Diabetes Data

A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data set was collected by the US National Institute of Diabetes and Digestive and Kidney Diseases (Smith et al, 1988). It contains eight attributes. Interested readers please refer to Appendix A.5 for more details of Pima Indians Diabetes data set.

A population size of 20 was used and the algorithm was run for 40 generations. Each chromosome in the population consisted of 8 ConG and 9 CoeG. Each of the 8 ConG was used to select or deselect one of the 8 attributes of the Pima Indians diabetes data. 8 CoeG represented the 8 tolerance values for the 8 attributes, and 1 CoeG represented the vigilance value. The GA also made use of $(\mu + \lambda)$ selection, XHC, and dynamic mutation. A few results were obtained by tuning the crossover and mutation probabilities of the GA.

8.5.1 Performance of GA-KFLANN with Different Mutation Probabilities

Different mutation probabilities were used to find the best setting for GA-KFLANN.

Figure 8.16 shows the results of GA-KFLANN with different mutation probabilities.

Figure 8.16 Plots of Average Accuracy Obtained Using Different Mutation Probabilities

Slight difference can be observed from the plots in Figure 8.16. The plots with $P_m = 0.2$ and 0.5 converge a bit late from generation 32 and 33 respectively. The other with $P_m = 0.05$, converging from generation 29, gave more time for initial global search as well as local hill-climbing later on. $P_m = 0.05$ is chosen as the proper setting for clustering Pima Indians Diabetes data using GA-KFLANN.

8.5.2 Performance of GA-KFLANN with Different Crossover Probabilities

Different crossover probabilities were used to find the best setting for GA-KFLANN.

Figure 8.17 shows the results of GA-KFLANN with different crossover probabilities.

Figure 8.17 Plots of Average Accuracy Obtained Using Different Crossover Probabilities

From Figure 8.17, the plot with $P_c = 0.45$ shows premature convergence of the population of GA-KFLANN, while the plot with $P_c = 0.95$ show the lack of local search in the later generations. The setting with $P_c = 0.85$ gave the best solution to the generations for global search and local search. Therefore, $P_c = 0.85$ is chosen as the proper setting for clustering using GA-KFLANN in Pima Indians Diabetes data.

From the above testing, it is determined that the proper settings for GA-KFLANN on Pima Indians Diabetes data will be $(\mu + \lambda)$ selection, XHC, dynamic mutation with $P_m = 0.05$, and $P_c = 0.85$.

8.5.3 Comparison with Other Methods

As shown in Table 8.5, GA-KFLANN was also able to obtain 4.1% higher accuracy than K-NN and reduce the dimension from eight to three.

Table 8.5 Comparison of Clustering Results on Pima Indians Diabetes Data

	Dimension	Accuracy
GA-KFLANN	3	76%
K-NN	Full Set	71.9%

8.6 Comparison with Exhaustive Search Algorithm

ES-KFLANN was tested on three data sets and comparisons were made in three aspects: dimension reduction, clustering accuracy, and completion time.

Table 8.6 Comparison of GA-KFLANN and ES-KFLANN on Feature Selection

	Iris	Wine	WBC
GA-KFLANN	1	6	4
ES-KFLANN	1	3	5

Table 8.7 Comparison of GA-KFLANN and ES-KFLANN on Clustering Accuracy

	Iris	Wine	WBC
GA-KFLANN	98%	96%	96%
ES-KFLANN	98%	95.51%	96.5%

Table 8.8 Comparison of GA-KFLANN and ES-KFLANN on Completion Time

	Iris	Wine	WBC
GA-KFLANN	20 sec	1 min 3 sec	5 min 36 sec
ES-KFLANN	5 min	> 2 weeks	> 2 weeks

As shown in the above three tables, it is clear that the performance of GA-KFLANN is as good as ES-KFLANN in the field of feature reduction and accurate clustering. Besides, GA-KFLANN is much faster than ES-KFLANN in clustering data sets. Although ES-KFLANN is supposed to obtain the highest accuracy and most significant factors from data sets given enough time, GA-KFLANN is able to achieve comparable results within much less time. Therefore, GA-KFLANN is more efficient and practical than ES-KFLANN in data mining.

8.7 Overall Comparison

Table 8.9 shows the clustering results obtained using four different algorithms. Figure 8.18 was plotted according to the data in Table 8.9 in order to show the comparison more clearly.

Table 8.9 Comparison between GA-KFLANN and other Algorithms

	GA-KFLANN	ES-KFLANN	SD-KFLANN	K-NN
Iris	98%	98%	100%	95.1%
Wine	96%	95.51%	90.77%	96.7%
WBC	96%	96.5%	91.77%	96.6%
Glass	77%	-	69.05%	72%
Pima	76%	-	-	71.9%

Figure 8.18 Clustering Accuracy Obtained Using GA-KFLANN, SD-KFLANN, and K-NN

It is shown that GA-KFLANN has higher clustering accuracy than SD-KFLANN on almost every data set (except Iris) used in testing. Comparing with K-NN, GA-KFLANN has slightly lower accuracy on wine and WBC data, while it obtained better results on Iris, Glass, and Pima data. Therefore, it can be concluded that GA-KFLANN is able to achieve accurate results in data clustering.

According to the experiments and discussions above, GA-KFLANN is able to perform efficient and effective search with much less time consuming comparing with ES-KFLANN. It is also able to achieve comparably accurate clustering comparing with other techniques. Therefore, introducing GA into KFLANN indeed improves the overall performance of the original KFLANN algorithm and data mining using the hybrid technique is feasible and practical.

Chapter 9 Conclusions

KFLANN has the ability to cluster effectively, with consistent centroids, regardless of variations in the data presentation sequence as long as the correct settings are used for the two network parameters: tolerance δ and vigilance ρ . However, it is very hard to determine the proper values for δ and ρ without the domain knowledge of the data to be clustered, because the search space of the network parameters is complex and multi-modal. Several search techniques have been implemented in order to solve this problem. Some of them require the domain knowledge that is unknown in most cases, while some can performed fast clustering with crude results, and others achieving decent accuracy suffer from the curse of dimensionality. And in ES-KFLANN, an exhaustive search method was employed to explore all the possible combinations of tolerance and vigilance values. According to the clustering results on three test data sets shown previously, its search time on tolerance δ and vigilance ρ for clustering increases exponentially compared to the linear increase in the input dimension. It was feasible but not practical and efficient.

Genetic Algorithms (GAs) are guided by the population's fitness, yet somewhat left to the mercy of random search algorithms for complex optimization problems which are based on principles from natural evolutionary theory. The ability of GAs to search through the complex parameter space and likeliness to reach a near-optimal condition regardless of dimensionality is the strong motivation for introducing GAs into KFLANN. Therefore,

the hybrid model of GA-KFLANN was proposed. A GA was used to efficiently orchestrate the search of tolerance and vigilance values and perform feature selection to extract significant factors out of the data.

In the experiments, five data sets from well-known resources were used to investigate the performance and feasibility of GA-KFLANN. Different settings for the GA were also tested to achieve the best solutions for GA-KFLANN. These tests suggested that $(\mu + \lambda)$ selection, crossover hill-climbing and dynamic mutation would generally improve the performance of GA-KFLANN in clustering any data set, while the mutation and crossover probabilities could vary for different data sets. The test results of clustering Iris, wine and Wisconsin breast cancer (WBC) data showed that GA-KFLANN achieved competent accuracy in much less time comparing with the exhaustive search employed in the last research project. KFLANN using standard deviation also generated less accurate results comparing with GA-KFLANN on 3 out of 4 data sets tested. And comparing with K-NN, GA-KFLANN had slightly lower accuracy on wine and WBC data, while it obtained better results on Iris, glass, and pima Indians diabetes data.

From the discussions and experimental results provided in Chapters 7 and 8, it can be concluded that GA-KFLANN is able to perform efficient and effective search in a less time consuming manner as compared with ES-KFLANN. Furthermore, it is able to achieve comparably accurate clustering results. Introducing GA into KFLANN improves the overall performance of the original KFLANN algorithm and is able to find the

suitable settings for tolerance and vigilance values. Therefore, the original KFLANN algorithm is successfully adapted for efficient and effective data mining using GA.

However, GA-KFLANN did not achieve the highest accuracy on all the data sets used in the experiment comparing with SD-KFLANN and K-NN from Figure 8.18. The reason can be that GA is a random search technique. It can provide sub-optimal solutions to complex optimization problems, but does not guarantee that an optimal solution can be obtained. Therefore, an adaptive algorithm with the ability to seek out the optimal settings for both network parameters of KFLANN can be investigated in the future work.

Reference

- [1] B. Burdsall and C. Giraud-Carrier. GA-RBF: A Self-Optimising RBF Network. Proceedings of the Third International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA'97), pp 348--351. Springer-Verlag, April 1997.
- [2] B. Everitt, *Cluster Analysis*, 2nd ed., New York, Halsted Press, 1980.
- [3] B. Ster and A. Dobnikar, Neural Networks in Medical Diagnosis: Comparison with Other Methods. In A. Bulsari et al., editor, Proceedings of the International Conference EANN '96, pp 427-430, 1996.
- [4] C. M. Bishop, *Neural Networks for Pattern Recognition*, Clarendon Press, 1995.
- [5] C.L. Blake and C.J. Merz, UCI Repository of machine learning databases [<http://www.ics.uci.edu/~mllearn/MLRepository.html>], Irvine, CA: University of California, Department of Information and Computer Science, 1998.
- [6] C. Stergiou and D. Siganos, "Neural Networks", SURPRISE 96 Journal, Imperial College of Science Technology and Medicine, Vol. 4, 1996.
- [7] C.R. Rao, *Linear Statistical Inference and Its Application*, John Wiley & Sons, pp. 582-585, 1973.
- [8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company, Inc., California, 1989.
- [9] D. J. Evans and L. P. Tay, "Fast Learning Artificial Neural Networks for Continuous Input Applications", *Kybernetes*, Vol. 24, No. 3, 1995.
- [10] D. J. Hand, *Discrimination and Classification*, New York: John Wiley, 1981.

- [11] Gad, Esam E., Neural Network Evolution Using Genetic Algorithms: Pattern Classification and Logic Gate Recognition, UMI, 1991.
- [12] G. H. Dunteman, Principal Components Analysis, Newbury Park, Sage Publications, 1989.
- [13] I. T. Jolliffe, Principal Component Analysis, New York: Springer-Verlag, 1986.
- [14] J. Hartigan, Clustering Algorithms, New York: John Wiley & Sons, 1975.
- [15] J. M. Buhmann, "Data Clustering and Learning", The Handbook of Brain Theory and Neural Networks, 2nd ed., Cambridge, MA, The MIT Press, 2002.
- [16] J. W. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, San Francisco, 2001.
- [17] J. Yang, R. Parekh, and V. Honavar, "DistAL: An interpattern distance-based constructive learning algorithm", *Intelligent Data Analysis*, vol. 3, pp. 55--73, 1999.
- [18] J. Yang and V. Honavar, "Feature Subset Selection Using a Genetic Algorithm", *IEEE Intelligent Systems*, 13(2):44-49, 1998.
- [19] Leng Phuan Alex Tay, Lai Ping Wong, Jian Xu, "The K-Means Fast Learning Artificial Neural Network (KFLANN) - A Network for Consistent Cluster Formations", *Internal Report - ER-LAB_AUGUST_2005_01*, Nanyang Technological University.
- [20] L. P. Tay and D. J. Evans, "Fast Learning Artificial Neural Network (FLANN II) Using Nearest Neighbour Recall", *Neural Parallel and Scientific Computations*, Vol. 2, No. 1, 1994.

- [21] L. P. Tay and S. Prakash, K-Means Fast Learning Artificial Neural Network, an Alternative Network for Classification, ICONIP, 2002.
- [22] L. P. Wong and L. P. Tay, Centroid Stability with K-Mean Fast Learning Artificial Neural Networks, IJCNN, Vol.2, pp 1517 – 1522, 2003.
- [23] L. P. Wong, J. Xu and L. P. Tay, Liquid Drop Photonic signal using Fast Learning Artificial Neural Network, ICICS, Vol.2, pp. 1018- 1022, 2003.
- [24] M. Lozano, F. Herrera, N. Krasnogor and D. Molina, Real-Coded Memetic Algorithms with Crossover Hill-Climbing, Evolutionary Computation, Vol. 12, pp. 273 – 302, Issue 3 - Special Issue on Memetic Algorithms, 2004.
- [25] M. Meila, Data centering in feature space, Technical Report 421, University of Washington, 2002.
- [26] P. M. Narendra, and K. Fukunaga, A Branch and Bound Algorithm for Feature Subset Selection, IEEE Transactions on Computers 26 (9), 917-922, 1977.
- [27] P. S. Churchland, and T. J. Sejnowski, The Computational Brain, Cambridge, MA: MIT Press, 1992.
- [28] R. A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems", *Annual Eugenics*, 7, Part II, 179-188, 1936.
- [29] R. Bellman, Adaptive Control Processes: A Guided Tour. New Jersey: Princeton University Press, 1961.
- [30] S. D. Bay, Combining Nearest Neighbor Classifiers through Multiple Feature Subsets, Proc. 17th Intl. Conf. on Machine Learning, pp. 37-45, Madison, WI, 1998.

- [31] S. Grossberg, “Competitive Learning: from Interactive activation to adaptive resonance”, *Cognitive Science*, 11, pp. 23-96, 1978.
- [32] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd Ed., Upper Saddle River, NJ, Prentice Hall, 1999.
- [33] StatSoft Inc., “Data Mining Techniques”, *Electronic Statistics Textbook*, <http://www.statsoftinc.com/textbook/stdatmin.html>, [Accessed 28th Jul, 2002].
- [34] S. R. Ladd, *Genetic Algorithms in C++*, New York, M&T Books, 1996.
- [35] T. Jones, “Crossover, Macromutation, and Population-Based Search”, In Eshelman, L., editor, *Proc. Of the Sixth Int. Conf. on Genetic Algorithms*, pages 73-80, Morgan Kaufmann, San Francisco, California, 1995.
- [36] T. Kohonen, *Self Organization and Associative Memory*, 3rd Edition, Springer-Verlag, Berlin, 1989.
- [37] U. Maulik and S. Bandyopadhyay, “Genetic Algorithm-Based Clustering Technique”, *Pattern Recognition*, Vol. 33, No. 9, pp. 1455-1465, 2000.
- [38] U.M. O'Reilly and F. Oppacher, “Hybridized Crossover-Based Search Techniques for Program Discovery”, *IEEE International Conference on Evolutionary Computation 1995*, pages 573-578, IEEE Press, Piscataway, New Jersey, 1995.
- [39] W.H. Wolberg and O.L. Mangasarian, “Multisurface Method of Pattern Separation or Medical Diagnosis Applied to Breast Cytology”, *Proceedings of the National Academy of Sciences*, U.S.A., Vol. 87, pp 9193-9196, 1990.

- [40] X. Yin and L. P. Tay, Feature Extraction Using The K-Means Fast Learning Artificial Neural Network, ICICS, Vol.2, pp. 1004- 1008, 2003.
- [41] X. Yin and L. P. Tay, “Genetic Algorithm Based K-Means Fast Learning Artificial Neural Network”, *17th Australian Computer Society Conference on Artificial Intelligence (Lecture Notes in Artificial Intelligence (LNAI))*, 2004.
- [42] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Verlag, 1992.

Appendix A

A.1. Iris Data

1. Title: Iris Plants Database

2. Sources:

(a) Creator: R.A. Fisher

(b) Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)

(c) Date: July, 1988

3. Past Usage:

- Publications: too many to mention!!! Here are a few.

1. Fisher, R.A. "The use of multiple measurements in taxonomic problems"

Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to Mathematical Statistics" (John Wiley, NY, 1950).

2. Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis.

(Q327.D83) John Wiley & Sons. ISBN 0-471-22361-1. See page 218.

3. Dasarthy, B.V. (1980) "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.

-- Results:

-- very low misclassification rates (0% for the setosa class)

4. Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule". IEEE Transactions on Information Theory, May 1972, 431-433.

-- Results:

-- very low misclassification rates again

5. See also: 1988 MLC Proceedings, 54-64. Cheeseman et al's AUTOCLASS II conceptual clustering system finds 3 classes in the data.

4. Relevant Information:

--- This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

--- Predicted attribute: class of iris plant.

--- This is an exceedingly simple domain.

5. Number of Instances: 150 (50 in each of three classes)

6. Number of Attributes: 4 numeric, predictive attributes and the class

7. Attribute Information:

1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. class:
 - Iris Setosa
 - Iris Versicolour
 - Iris Virginica

8. Missing Attribute Values: None

Summary Statistics:

	Min	Max	Mean	SD	Class Correlation
sepal length:	4.3	7.9	5.84	0.83	0.7826
sepal width:	2.0	4.4	3.05	0.43	-0.4194
petal length:	1.0	6.9	3.76	1.76	0.9490 (high!)
petal width:	0.1	2.5	1.20	0.76	0.9565 (high!)

9. Class Distribution: 33.3% for each of 3 classes.

A.2. Wine Data

1. Title of Database: Wine recognition data

2. Sources:

(a) Forina, M. et al, PARVUS - An Extendible Package for Data

Exploration, Classification and Correlation. Institute of Pharmaceutical

and Food Analysis and Technologies, Via Brigata Salerno,

16147 Genoa, Italy.

(b) Stefan Aeberhard, email: stefan@coral.cs.jcu.edu.au

(c) July 1991

3. Past Usage:

(1) S. Aeberhard, D. Coomans and O. de Vel,

Comparison of Classifiers in High Dimensional Settings,

Tech. Rep. no. 92-02, (1992), Dept. of Computer Science and Dept. of

Mathematics and Statistics, James Cook University of North Queensland.

(Also submitted to Technometrics).

The data was used with many others for comparing various

classifiers. The classes are separable, though only RDA

has achieved 100% correct classification.

(RDA : 100%, QDA 99.4%, LDA 98.9%, 1NN 96.1% (z-transformed data))

(All results using the leave-one-out technique)

In a classification context, this is a well posed problem with "well behaved" class structures. A good data set for first testing of a new classifier, but not very challenging.

(2) S. Aeberhard, D. Coomans and O. de Vel,
"THE CLASSIFICATION PERFORMANCE OF RDA"
Tech. Rep. no. 92-01, (1992), Dept. of Computer Science and Dept. of
Mathematics and Statistics, James Cook University of North Queensland.
(Also submitted to Journal of Chemometrics).

Here, the data was used to illustrate the superior performance of
the use of a new appreciation function with RDA.

4. Relevant Information:

-- These data are the results of a chemical analysis of
wines grown in the same region in Italy but derived from three
different cultivars.

The analysis determined the quantities of 13 constituents
found in each of the three types of wines.

-- I think that the initial data set had around 30 variables, but

for some reason I only have the 13 dimensional version.

I had a list of what the 30 or so variables were, but a.)

I lost it, and b.), I would not know which 13 variables are included in the set.

5. Number of Instances

class 1	59
---------	----

class 2	71
---------	----

class 3	48
---------	----

6. Number of Attributes

13

7. For Each Attribute:

All attributes are continuous

No statistics available, but suggest to standardise variables for certain uses (e.g. for us with classifiers which are NOT scale invariant)

NOTE: 1st attribute is class identifier (1-3)

8. Missing Attribute Values:

None

9. Class Distribution: number of instances per class

class 1	59
class 2	71
class 3	48

A.3. Wisconsin Breast Cancer Data

1. Title: Wisconsin Breast Cancer Database (January 8, 1991)

2. Sources:

-- Dr. William H. Wolberg (physician)

University of Wisconsin Hospitals

Madison, Wisconsin

USA

-- Donor: Olvi Mangasarian (mangasarian@cs.wisc.edu)

Received by David W. Aha (aha@cs.jhu.edu)

-- Date: 15 July 1992

3. Past Usage:

Attributes 2 through 10 have been used to represent instances.

Each instance has one of 2 possible classes: benign or malignant.

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

- Size of data set: only 369 instances (at that point in time)
- Collected classification results: 1 trial only
- Two pairs of parallel hyperplanes were found to be consistent with 50% of the data
 - Accuracy on remaining 50% of dataset: 93.5%
- Three pairs of parallel hyperplanes were found to be consistent with 67% of data
 - Accuracy on remaining 33% of dataset: 95.9%

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

- Size of data set: only 369 instances (at that point in time)
- Applied 4 instance-based learning algorithms
- Collected classification results averaged over 10 trials
- Best accuracy result:

- 1-nearest neighbor: 93.7%
- trained on 200 instances, tested on the other 169
- Also of interest:
 - Using only typical instances: 92.2% (storing only 23.1 instances)
 - trained on 200 instances, tested on the other 169

3) Aaron, K. (1998), Group Classification Using a Mix of Genetic Programming and Genetic Algorithms, In Proceedings of the 1998 ACM Symposium on Applied Computing (pp.308--312), Atlanta, GA.

4. Relevant Information:

Samples arrive periodically as Dr. Wolberg reports his clinical cases.

The database therefore reflects this chronological grouping of the data.

This grouping information appears immediately below, having been removed from the data itself:

Group 1: 367 instances (January 1989)

Group 2: 70 instances (October 1989)

Group 3: 31 instances (February 1990)

Group 4: 17 instances (April 1990)

Group 5: 48 instances (August 1990)

Group 6: 49 instances (Updated January 1991)

Group 7: 31 instances (June 1991)

Group 8: 86 instances (November 1991)

Total: 699 points (as of the donated database on 15 July 1992)

Note that the results summarized above in Past Usage refer to a dataset of size 369, while Group 1 has only 367 instances. This is because it originally contained 369 instances; 2 were removed. The following statements summarize changes to the original Group 1's set of data:

Group 1: 367 points: 200B 167M (January 1989)

Revised Jan 10, 1991: Replaced zero bare nuclei in 1080185 & 1187805

Revised Nov 22, 1991: Removed 765878,4,5,9,7,10,10,10,3,8,1 no record

Removed 484201,2,7,8,8,4,3,10,3,4,1 zero epithelial

Changed 0 to 1 in field 6 of sample 1219406

Changed 0 to 1 in field 8 of following sample:

1182404,2,3,1,1,1,2,0,1,1,1

5. Number of Instances: 699 (as of 15 July 1992)

6. Number of Attributes: 10 plus the class attribute

7. Attribute Information: (class attribute has been moved to last column)

#	Attribute	Domain

1.	Sample code number	id number
2.	Clump Thickness	1 - 10
3.	Uniformity of Cell Size	1 - 10
4.	Uniformity of Cell Shape	1 - 10
5.	Marginal Adhesion	1 - 10
6.	Single Epithelial Cell Size	1 - 10
7.	Bare Nuclei	1 - 10
8.	Bland Chromatin	1 - 10
9.	Normal Nucleoli	1 - 10
10.	Mitoses	1 - 10
11.	Class:	(2 for benign, 4 for malignant)

8. Missing attribute values: 16

There are 16 instances in Groups 1 to 6 that contain a single missing (i.e., unavailable) attribute value, now denoted by "?".

9. Class distribution:

Benign: 458 (65.5%)

Malignant: 241 (34.5%)

A.4. Glass Data

1. Title: Glass Identification Database

2. Sources:

(a) Creator: B. German

-- Central Research Establishment

Home Office Forensic Science Service

Aldermaston, Reading, Berkshire RG7 4PN

(b) Donor: Vina Spiehler, Ph.D., DABFT

Diagnostic Products Corporation

(213) 776-0180 (ext 3014)

(c) Date: September, 1987

3. Past Usage:

-- Rule Induction in Forensic Science

-- Ian W. Evett and Ernest J. Spiehler

-- Central Research Establishment

Home Office Forensic Science Service

Aldermaston, Reading, Berkshire RG7 4PN

-- Unknown technical note number (sorry, not listed here)

-- General Results: nearest neighbor held its own with respect to the rule-based system

4. Relevant Information:

Vina conducted a comparison test of her rule-based system, BEAGLE, the nearest-neighbor algorithm, and discriminant analysis. BEAGLE is a product available through VRS Consulting, Inc.; 4676 Admiralty Way, Suite 206; Marina Del Ray, CA 90292 (213) 827-7890 and FAX: -3189. In determining whether the glass was a type of "float" glass or not, the following results were obtained (# incorrect answers):

Type of Sample	Beagle	NN	DA
Windows that were float processed (87)	10	12	21
Windows that were not: (76)	19	16	22

The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence...if it is correctly identified!

5. Number of Instances: 214

6. Number of Attributes: 10 (including an Id#) plus the class attribute

-- all attributes are continuously valued

7. Attribute Information:

1. Id number: 1 to 214

2. RI: refractive index

3. Na: Sodium (unit measurement: weight percent in corresponding oxide, as
are attributes 4-10)

4. Mg: Magnesium

5. Al: Aluminum

6. Si: Silicon

7. K: Potassium

8. Ca: Calcium

9. Ba: Barium

10. Fe: Iron

11. Type of glass: (class attribute)

-- 1 building_windows_float_processed

-- 2 building_windows_non_float_processed

-- 3 vehicle_windows_float_processed

-- 4 vehicle_windows_non_float_processed (none in this database)

-- 5 containers

-- 6 tableware

-- 7 headlamps

8. Missing Attribute Values: None

Summary Statistics:

Attribute:	Min	Max	Mean	SD	Correlation with class
2. RI:	1.5112	1.5339	1.5184	0.0030	-0.1642
3. Na:	10.73	17.38	13.4079	0.8166	0.5030
4. Mg:	0	4.49	2.6845	1.4424	-0.7447
5. Al:	0.29	3.5	1.4449	0.4993	0.5988
6. Si:	69.81	75.41	72.6509	0.7745	0.1515
7. K:	0	6.21	0.4971	0.6522	-0.0100
8. Ca:	5.43	16.19	8.9570	1.4232	0.0007
9. Ba:	0	3.15	0.1750	0.4972	0.5751
10. Fe:	0	0.51	0.0570	0.0974	-0.1879

9. Class Distribution: (out of 214 total instances)

- 163 Window glass (building windows and vehicle windows)
 - 87 float processed
 - 70 building windows
 - 17 vehicle windows
 - 76 non-float processed
 - 76 building windows
 - 0 vehicle windows

-- 51 Non-window glass

-- 13 containers

-- 9 tableware

-- 29 headlamps

1,2,3,4,5,6,7.

id:ignore.

RI:continuous.

Na:continuous.

Mg:continuous.

Al:continuous.

Si:continuous.

K:continuous.

Ca:continuous.

Ba:continuous.

Fe:continuous.

A.5. Pima Indian Diabetes

1. Title: Pima Indians Diabetes Database

2. Sources:

(a) Original owners: National Institute of Diabetes and Digestive and

Kidney Diseases

(b) Donor of database: Vincent Sigillito (vgs@aplcn.apl.jhu.edu)

Research Center, RMI Group Leader

Applied Physics Laboratory

The Johns Hopkins University

Johns Hopkins Road

Laurel, MD 20707

(301) 953-6231

(c) Date received: 9 May 1990

3. Past Usage:

1. Smith, J.W., Everhart, J.E., Dickson, W.C., Knowler, W.C., \& Johannes, R.S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In Proceedings of the Symposium on Computer Applications and Medical Care (pp. 261--265). IEEE Computer Society Press.

The diagnostic, binary-valued variable investigated is whether the patient shows signs of diabetes according to World Health Organization criteria (i.e., if the 2 hour post-load plasma glucose was at least 200 mg/dl at any survey examination or if found during routine medical care). The population lives near Phoenix, Arizona, USA.

Results: Their ADAP algorithm makes a real-valued prediction between 0 and 1. This was transformed into a binary decision using a cutoff of 0.448. Using 576 training instances, the sensitivity and specificity of their algorithm was 76% on the remaining 192 instances.

4. Relevant Information:

Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage. ADAP is an adaptive learning routine that generates and executes digital analogs of perceptron-like devices. It is a unique algorithm; see the paper for details.

5. Number of Instances: 768

6. Number of Attributes: 8 plus class

7. For Each Attribute: (all numeric-valued)

1. Number of times pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)

5. 2-Hour serum insulin (μ U/ml)
6. Body mass index (weight in kg/(height in m)²)
7. Diabetes pedigree function
8. Age (years)
9. Class variable (0 or 1)

8. Missing Attribute Values: None

9. Class Distribution: (class value 1 is interpreted as "tested positive for diabetes")

Class Value	Number of instances
-------------	---------------------

0	500
---	-----

1	268
---	-----

10. Brief statistical analysis:

Attribute number:	Mean:	Standard Deviation:
-------------------	-------	---------------------

1.	3.8	3.4
----	-----	-----

2.	120.9	32.0
----	-------	------

3.	69.1	19.4
----	------	------

4.	20.5	16.0
----	------	------

5.	79.8	115.2
----	------	-------

6.	32.0	7.9
----	------	-----

7.	0.5	0.3
8.	33.2	11.8