Evaluation of a machine learning tool in the domain of genetic host responses

Joel Pitt, 0038830

Supervisor Brent Martin

November 7, 2003

This project is dedicated to my Oma, Marthea van der Reyden, who passed away peacefully at home after a sudden fight with lung cancer.

Acknowledgements

I would like to thank Brent Martin for being my supervisor over the past year, he was friendly and our conversations helped me formulate new ideas. Also I must thank Ben Goertzel the founder of Novamente who proposed the topic and put in me in touch with the Novamente Team.

Thank you to Lúcio de Souza Coelho, who was the expert on the classification system within Novamente, for never failing to cheerfully answer my questions despite being incredibly busy with work. He also informed me of many interesting Brazillian sayings.

Abstract

This report examines the use of Support Vector Machines and Genetic Programming Classifiers (GPCs) to distinguish between classes of cancer based on gene expression data. The effect of feature selection on classifier accuracy and on the convergence time of GPCs is experimentally investigated, with the goal of making classification problems on gene expression data tractable to GPCs.

Contents

Chapter 1

Introduction

Throughout biology today scientists are being deluged in data, and they are unable keep up with this constant stream without specialised tools. For this reason computational methods and computer software are being developed to help biologists cope and come to grips with the quantity of information available to them. These methods and their study are part of the expanding field of Bioinformatics, also known as Computational Biology, and has been vital in making sense of all the biological data available.

Originally this excess of data mainly came from sources such as DNA and protein sequencing, but recently microarray technology has become more common place. Microarrays are used to detect the expression levels of thousands of genes within cells, allowing biologists to extrapolate gene interactions and how these genes may regulate one another. The advent of this technology has brought to data analysts broad patterns of gene expression simultaneously recorded in a single experiment(Fodor 1997).

Another use of these gene expression data is in the domain of tissue classification. Although all cells within an organism contain the same set of genes, not all of these are expressed in all tissues and this is an important behaviour for tissue differentiation. Therefore, the pattern of genes expressed in a particular tissue sample should allow us to categorise the samples site of origin.

Cancer can arise from the improper expression of genes. such as genes involved in cell death (apoptosis) being switched off, or over expression of genes involved in cell proliferation. Thus the pattern of genes expressed by cancerous tissue will be altered from that of normal tissue and detection of these alterations by machine learning categorisation techniques allow us to classify between cancererous and normal tissue.

A further issue is being able to differentiate between cancerous tissue. For example, given a cancerous tissue sample, can we determine its tissue of origin? Since the gene expression patterns between normal tissues are different, we can assume that tumours originating from these normal tissues will also exhibit different gene expression patterns. Although there are likely to be certain genes that show a commonality in their expression among many tumours. Being

able to differentiate between cancerous tissue is useful when a tumour is first discovered within a patient, because it is initially unknown whether this tumour originated in the tissue it was found in or is a secondary tumour resulting from a primary tumour metastasis.

Microarray data is often made available to the public online in a manner similar to the human genome project. A dataset of 280 human tissue samples that under went microarray processing was retrieved from (Ramaswamy 2001). These tissue samples were taken from both healthy individuals and from tumours in individuals suffering cancer. The original paper was interested in classifying the data using Support Vector Machines (SVMs) within a One Vs. All (OVA) scheme.

The aim of this project is to make use of Support Vector Machines (SVMs), a machine learning classifier, to classify between cancer types and normal tissue. SVMs are very good at what they do, but are unintuitive to Novamente, an artificial intelligence engine designed to show the emergence of true general artificial intelligence. The way SVMs store the classification model prevents Novamente from being able to (easily) draw inference from the learning of cancer classes. We propose to determine what features of the cancers are relevent via SVMs and Recursive Feature Elimination, then use these features in Genetic Programming predictors. Genetic Programming results in rules much closer to the underlying architecture of Novamente. A comparison of the accuracy between SVMs and genetic algorithms using the same set of features will be made.

1.1 Overview

Further background material is presented in Chapter 2 where cancer and microarrays are discussed, as well as details on Support Vector Machines and Genetic Programming. Chapter 3 specifies the technical details of the experiments carried out for this report, with the results presented in Chapter 4. Chapter 5 provides a discussion of some interesting points, and further work to be done in the area of classification from gene expression data. Finally, Chapter 6, concludes the report.

Chapter 2

Background

Currently the diagnosis of cancer involves the interpretation of clinical and histopathological data with the goal of ultimately placing a tumour into one of the currently accepted categories. There are a number difficulties here. For instance, there is a wide spectrum of cancer morphology and many tumours are atypical, lacking morphologic features often used in diagnosis (Ramaswamy et al. 2001). To this effect there have been calls for mandatory second opinions in all surgical pathology cases (Tomaszewski & LiVolsi 1999).

These difficulties hinder our ability to give patients the optimum care possible as well as increasing the expense of caring for them. Differences between tumours classified similarly by doctors may even confound clinical trials for new anti-cancer treatments.

Molecular diagnostics promise precise objective classification of tumours. However, characteristic molecular markers have yet to be identified for most tumour types (Connolly et al. 1997). Gene Expression diagnostics have the advantage that expressions of genes with unknown functions can be monitored.

Genes important to the development of cancer are potential places to look for creating molecular markers. They may encode for altered proteins specific to a cancer class and molecular diagnostic markers based on these proteins can be made.

Ramaswamy ((2001)) explored the possibilty of creating a reference database of tumour gene expression data for all common malignancies. This could then serve as test bed for testing new cancer classification methods. Ramaswamy tried several machine learning methods on the initial version of their reference database, including k-Nearest Neighbour, Weighted Voting and Support Vector Machines. They also investigated using unsupervised clustering.

2.1 Microarrays

A microarray allows a researcher to view the expression levels of thousands of genes simultaneously at a particular point in time by taking advantage of an

intermediary step in the expression of genes. At this step, DNA is transcribed into messenger RNA (mRNA) before being transported to ribosomes where it is translated into a protein, which carries out the function of the gene.

In order to construct a microarray, precisely measured quantities of single-stranded cDNA (copy DNA) transcribed from mRNA are bound to spots on a glass slide. These slides are then rinsed with fluorescently labelled mRNA from the cells whose expression levels wish to be observed. The strands of fluorescent mRNA that are complementary to those on the slide will bind, and when the slide is developed the brightness of individual spots indicate a quantitative measure of how much of each mRNA strand was present in the cells at the moment of sampling. This measure indicates how much the gene is being expressed - the more mRNA molecules that are transcribed from a particular gene the more protein that can be translated.

In essence this means a microarray provides a snapshot of all the genes that are of interest and their levels of expression. The collated data from a set of microarray experiment form a *gene expression matrix* where each column represents a tissue sample and each column refers to a gene.

2.2 Novamente

Novamente is an artificial intelligence (AI) engine that has the lofty goal of general artificial intelligence (GAI), developed by the Artificial General Intelligence Research Institute (AGIRI)[1]. Thus, it differs from other AI engines in that it will be capable of being applied to any domain and will be able to formulate its own goals and beliefs. The creators behind this engine are fully aware of the failed attempts of other similar projects, but believe that the fundamental design of the Novamente mind will result in emergent intelligence (Goertzel et al. 2003+).

This project does not rely on any of the explicit GAI properties of the Novamente engine, so that belief in whether GAI is possible or not will not affect Novamente's use as a powerful machine-learning tool. Novamente combines numerous AI paradigms, both symbolic and connectionist views, allowing it to exhibit traditional domain-specific AI behavior.

One of the most promising points about Novamente in relation to bioinformatics, is that it can incorporate background knowledge from biological databases when datamining. Such behaviour is still in a state of development within Novamente, so it was decided to forego using this background knowledge during experiments.

The basic architecture of the Novamente AI engine is similar to a neural network in that it is composed of numerous nodes and the relationships between them. These nodes and relationships can then be acted upon by *MindAgents* that carry out the system dynamics. Some nodes contain instructions for carrying out more complicated actions and these instructions are executed by MindAgents.

[1] http://www.agiri.org

Novamente was a progression from a previous but similar engine called Webmind. Due to commericial reasons the company behind Webmind dissolved, but the core developers went on to develop Novamente for AGIRI. Novamente was designed from Webmind, but significant changes were made made based on the teams experience. For example, Webmind was programmed in Java, whereas Novamente is implemented in C++.

2.3 Machine Learning

The term *learning* can embody a wide scope of processes. Dictionary definitions often emphasize the acquisition of skill and knowledge and the associated cognitive processes. Learning is most often prescribed to living creatures when their behaviour changes based on past experience. However, the same term can (and is) used to describe machine behaviour when the machine takes into account past experience to act in the present.

> "[A] machine learns whenever it changes its structure, program, or data (based on its inputs or in response to external information) in such a manner that its expected future performance improves."(Nilsson 1996)

Machine learning, like the term it is named after, also embodies a wide scope of processes. Here we a primarily interested in *classification*.

2.3.1 Classification

Classification is the process of dividing *instances* into *classes* according to the values of an instance's features. Each sample occupies a point in *feature space* which is a theoretical N-dimensional space where each feature is uniquely assigned to an orthoganal axis (Kuravilla et al. 2002).

Figure 2.1 shows a feature space for samples with two features and hence two dimensions.

Unsupervised Classification

Unsupervised classification, otherwise known as *clustering*, attempts to group instances without reference to predetermined classes. This is often based on the proximity of the instances within feature space (Eisen et al. 1998). Clustering can be useful in identifying subtypes of cancer, and in detecting previously unrecognised similarities.

In order to carry out clustering, a *distance matrix* must be constructed where entry (x, y) is the distance between instances x and y. The entry values are determined from a *distance metric* and therefore choice of this metric can affect the outcome of clustering. The distance metric is often the Euclidean

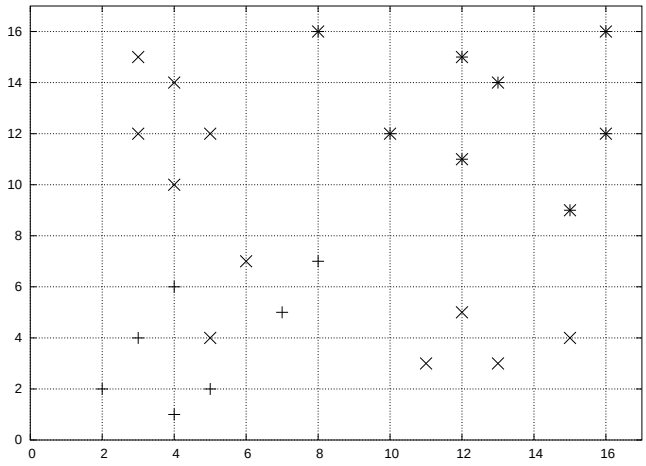


Figure 2.1: An example of a two dimensional feature space containing samples of three different classes.

distance[2] between points or the related Pearson correlation which is invariant to scaling.

Supervised Classification

Supervised Classification makes use of a training set containing instances for which the classes they belong to is known. After processing of this training set a class predictor is created for classifying unlabelled instances (Radmacher et al. 2002). For cancer this involves assigning the clinically defined tumour class to each gene expression profile in the training set.

2.3.2 Support Vector Machines

Support Vector Machines (SVMs) are powerful supervised classification systems based on a variation of regularization techniques for regression (Vapnik 1998) (Evgeniou et al. 2000). They provide extremely good performance in a wide range of binary classification domains with computational advantages over their contenders(Cristianini & Shawe-Taylor 1999). SVMs are a regularization of an older machine learning method called the *perceptron*(Rosenblatt 1962) (Minsky & Papert 1972). The perceptron tries to find a hyperplane that seperates positive from negative classes, and in general there may be any number of such hyperplanes. SVMs specify that this hyperplane must have the maximal *margin* (the distance from the hyperplane to the nearest point).

[2]The Euclidean distance equals the square root of the sum of the distances between points on each axis.

The creation of the hyperplane for the SVM is an optimisation problem, where the SVM must construct a hyperplane to maximise the distance from the hyperplane, W, and the closest instances to the hyperplane. This distance is calculated in N-dimensional euclidean space, where N corresponds to the number of features in each instance. Training an SVM and finding this hyperplane requires solving a convex quadratic problem with as many variables as training points. Kernels may be employed by SVMs to transform the data so that non-linear seperation can be captured. Different kernels or methods of transformation exist, the simplest being *linear* which performs no transformation. A *polynomial* kernel allows the SVM to capture any arbitary seperation by finding the order-N polynomial that best fits. Similarly for the *gaussian* kernel.

The class of an unknown test sample is determined by which side of the hyperplane it lies on. It is given the same class as the training samples on the side of the hyperplane which it is located on.

SVMs have been applied to several biological domains, including gene expression microarrays (Mukherjee 1999)(Brown et al. 2000) and are particularly suited to gene expression based categorisation since they can handle the large number of features often produced in this domain.

For further information on Support Vector Machines, including mathematical proofs, see either (Vapnik 1998) or (Evgeniou et al. 2000).

2.3.3 Genetic Programming

Genetic Classifiers come under the broader term of Evolutionary Algorithms (EAs) which was popularised by (Holland 1975) and is inspired by Darwin's evolutionary theory (Darwin 1859).

> "[All the fields within evolutionary computing] share a common conceptual base of simulating the evolution of individual structures via processes of selection, mutation, and reproduction. The processes depend on the perceived performance of the individual structures as defined by an environment."[3]

EAs can be thought of as an optimisation problem, searching through potential solutions to find the optimal one (Michalewicz 1996)

The population contains initially poor and randomly created solutions to the problem. These are then evaluated using a fitness function, with the fittest selected and mated to create a new population. To create offspring (problem solutions) for the new population, the best solutions of the current population are subjected to *crossover recombination*. These solutions can be thought of as points of a landscape called the solution space, where each point has a certain fitness associated with it. Selecting the fittest individuals push these points "uphill" to areas of higher fitness (Kauffman 1995). Mutation can also occur to introduce further variation and prevent over crowding of a solution space. This

[3]The Hitch-Hiker's Guide to Evolutionary Computation (FAQ for comp.ai.genetic)

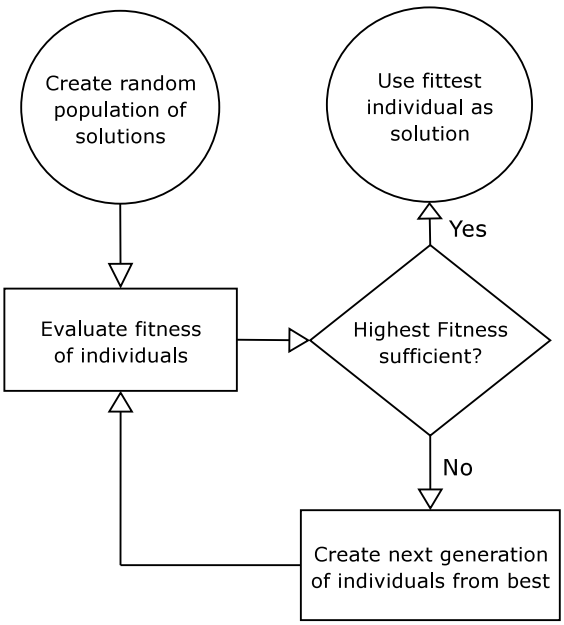


Figure 2.2: Flow chart of the evolutionary algorithm process.

also helps avoid a population becoming stuck near local optima and introduces attributes that may not have been present in the initial population.

Recombination and mutation may result in offspring with superior fitness over their ancestors. A regime following the often coined term "survival of the fittest" ensures the best solutions, or parts of them, progress to the next generation. Through iterated generations better solutions are repeatedly found and those attributes which contribute to a high fitness accumulate. This evolution however is bounded by the ideal solution to the problem, or the global optimum in the solution space.

Figure 2.2 shows a flow chart of the evolutionary algorithm process. The population is initially created, all individuals then have their fitness evaluated and then it is compared to the required fitness. If the fitness of the best individual meets this minimum required fitness then the process ends and the individual is returned as a result of the evolutionary algorithm. Otherwise the best individual(s) is taken and used to generate a new population which then allows the process to repeat. Sometimes the stopping condition is replaced with a counter for the number of generations that have passed.

Sometimes evolutionary process are misunderstood as a purely random search through the solution space, which leads to justifiable doubt over the usefulness of evolution. There is obviously random aspects in evolutionary processes, such as mutation and the location at which recombination occurs, but the important point is that attributes representing high fitness are *accumulated* over genera-

tions. Offspring are created based on already proven attributes (Dawkins 1996). Ultimately, the definition of fitness and the fitness function, whether artificial or natural, are important in guiding the progression of evolutionary processes.

Novamente implements a classifier system that uses genetic programming to create individual solutions (each a function tree) capable of differentiating between classes. These are known as Genetic Programming Classifiers (GPCs).

Fitness Function

The fitness function determines how closely a candidate matches the ideal solution. Generally the fitness function is a rank based fitness assignment (Baker 1985), or a proportion of the average fitness of the population (Goldberg 1989).

As generations pass, the best solution converges towards the ideal. This is reflected in the fitness difference between the fittest candidate and the ideal solution approaching zero.

Novamente uses a fitness function based on the number of incorrectly classified training examples:

$$fitness = \frac{1}{1+n} \tag{2.1}$$

$$\tag{2.2}$$

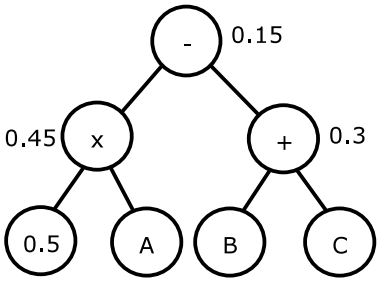
Where n is the number of incorrectly classified examples. This limits the fitness of a candidate in the range $[0, 1]$, with a fitness of 1 representing the ideal solution: a classifier that makes no mistakes in classifying the training data.

Population Size

The population size is the number of individual solutions present during any particular generation. This is usually kept constant over all generations for simplicity. If constant population size is maintained, then the number of new individuals created during each new generation is dependant on the survival of the offspring's parents.

Some evolutionary methods allow individuals to mate to create new individuals for the next generation, before being "killed". While others select a percentage of individuals with high fitnesses and place them in the next generation, filling the remaining spaces with their offspring. Allow parents to survive limits the amount of available positions for offspring, or new solutions to the problem, and can result in a population being caught on local optima (Kauffman 1995) when parents become too similar. Similarity prevents cross-over from creating sufficiently unique individuals to explore alternative solutions. However, allowing parents to survive can avoid oscillation of the population's maximum fitness which could lead to a final solution being less fit then it could have been.

Novamente uses a model in which certain randomly selected individuals in a subgroup compete and the fittest individual is selected. This is called tournament selection and is the form of selection we see in nature. For instance, stags



Gene	Value
A	0.9
B	0.2
C	0.1

Figure 2.3: Each individual in a population of genetic programming solutions is a function tree. The tree's leaf nodes take inputs from the data and can contain constants, the rest of the nodes contain operators which recieve input from their children nodes. Evaluating a function tree results in a value at the root node (in this case 0.15). During training this value is compared to the desired value in order to calculate the individual's fitness.

rut to vie for the privilege of mating with a herd of hinds, and the winner gets to pass their genes to the following generation.

Novamente also employs the concept of *elitism*. This is where the fittest individual from a generation is allowed to live on in the next generation. This combats possible oscillations in maximum fitness, but prevents being caught in local optima as the offspring are free to pursue alternative solutions.

Function Trees

Each individual in a genetic programming system represents a binary *function tree* which is much like the trees created by some rule-based classification systems. Each node in the tree can contain an operator from a predetermined set (such as multiply, add, maximum, minimum etc.), recieve an input from a specific feature in incoming examples, or be a constant value (see Figure 2.3).

Traditionally cross-over recombination and mutation have been defined on a linear chromosome (due to the biological origins of these processes), and it is not immediately clear how these procedure would occur on a tree. Tree cross-over is facilitated by randomly selecting a node within the one tree and exchanging it, and its children nodes, with a randomly selected node in another tree.

A mutation on a tree selects a random node and either changes the operator, the feature to take input from, or its value if it is a constant.

When using genetic programs for classification, each function tree created represents a class that it is being evolved to recognise. A tree is evaluated against an example by setting all of its input nodes to the values of the specified features

in the example. Each node then performs its operation, taking input from its children. If the children also have operations associated with them then they are first evaluated. This occurs until nodes with either feature values or constants are reached (these are always leaf nodes), then the values are processed by their parents and the results propagate up through the tree until ultimately the root node is evaluated. The value that the root node outputs by performing its operation is the final output of the tree (unless the entire tree only consists of the root node, in which case either the direct value of a feature or a constant value is returned). This output is limited to the range $[0 \ldots 1]$, with 1 representing a positive classification and 0 a negative one.

Usually the depth of the trees are predetermined, and this affects the number of feature inputs that can be incorporated into the classifier. For example, if we have examples with 500 features, then we would need at least 500 leaf nodes in the tree. Since we are dealing with binary trees the minimum depth can be expressed as:

$$depth \quad = \quad ceil(\frac{\log N}{\log 2}) \tag{2.3}$$

where N is the number of features, and the function $ceil(x)$ returns the first integer greater than x. For 500 features this works out to a depth of 9.

The tree depth has a major impact on training speed since each tree needs to be evaluated for its fitness and each increase in depth doubles the number of leaf nodes. This increases the number of operations needed for tree evualation and the relationship can be seen in Figure 2.4

$$operations \quad = \quad 2^{d-2} + 2^{d-3} + ... + 2^0 \tag{2.4}$$

Equation 2.4 determines the number of operations a function tree performs, where d is the depth of the tree. The series starts at 2^{d-2} because the first depth is represented by 2^0 and the 2^{d-1} leaf nodes are inputs rather than operators. Equation 2.4 simplifies to

$$operations \quad = \quad 2^{d-1} - 1 \tag{2.5}$$

Convergence

Convergence is process of an individual's fitness reaching its highest possible value given the constraints of the function trees. Often the optimal solution and its associated fitness is unknown however. If the highest fitness for a population remains stable for large number of generations it is assumed to be the optimal solution. However, it may only be a local optima, preventing the population from exploring alternative possibilities.

Genetic program classifiers are adversely affected by excessive irrelevent features, because a mutation on an input node is less likely to attach it to a useful

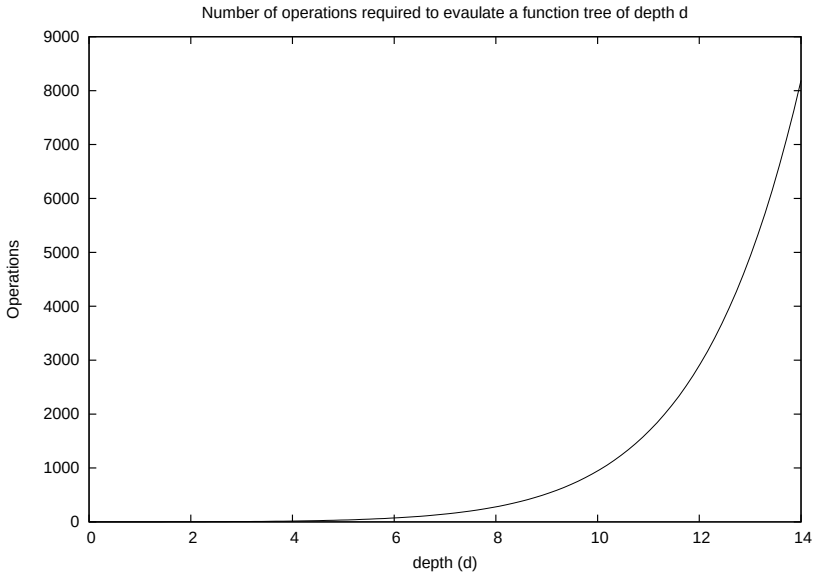


Figure 2.4: The number of operations required to evaulate a function tree of depth d.

feature. This increases the number if mutations required before a relevent feature is selected and therefore the number of generations to train a classifier to a certain accuracy also increases.

This is a important problem when dealing with gene expression data, because each sample often contains thousands of features, most of which are not associated with the classes that are trying to be differentiated. Feature selection can reduce the amount of irrelevent features and reduce the amount of time required to train a genetic program classifier to a tractable amount.

2.3.4 Feature Selection

Feature selection is the process of removing features from a dataset while trying to maintain the highest possible level of classification accuracy. There are several reasons to conduct such a process, one of which was discussed in relation to GPCs in section 2.3.3. Another, is in an attempt to overcome *overfitting*.

Overfitting occurs when the number of features is large while the number of training examples is comparatively small, and produces classifiers that can accurately classify the training examples, but perform poorly on real data. It is essentially the classifier matching the training data too well, such that will not recognise examples that have slightly differences. Overfitting can be overcome by either using the training techinique of regularisation (Vapnik 1998), or reducing the dimensionality of the data. The latter of which is used in this project.

To reduce the dimensionality of the data, we have several options. Firstly we can *project* the features onto a few principal dimensions. In this way new features are obtained that are linear combinations of original features (Duda & Hart 1973). This method however has the disadvantage that none of the original features are discarded. Secondly, we can employ one of a number of *pruning* methods which eliminate features from the data set. Pruning has practical importance to diagnostic tests for cancer (or other diseases) as it improves cost effectiveness of searching for marker sites, and eases verification of gene relevence. Pruning is what this report refers to when feature selection is mentioned. See (Kohavi & John n.d.) for a review of different feature selection methods.

If we have a small number of features, it is conceivable for us to exhaustively search through all subsets of features and evaluate each model bsed on selection criterion (Kearns et al. 1997). This is of course impractical for gene expression datasets with thousands of features due to the combinatorial explosion of subsets.

Feature Selection by Correlation Coefficient

Features can be invididually selected based on how much they contribute to the seperation of classes can produce a simple feature ranking. Golub et al. used a correlation coefficient defined as:

$$w_i = \frac{(\mu_i(+) - \mu_i(-))}{(\sigma_i(+) + \sigma_i(-))} \tag{2.6}$$

where μ_i and σ_i are the mean and standard deviation of the gene expression values of gene i for all instances of the positive (+) and negative (-) classes, $i = 1..n$. This results in large positive values for genes correlated with the positive class and large negative values for those correlated with the negative class. Golub et al. used this coefficient to select equal amounts of positive and negatively correlated genes during selection. Since then the absolute value of w_i has been used as a ranking criterion (Furey et al. 2000) and Pavlidis et al. have used a similar coefficient:

$$w_i = \frac{(\mu_i(+) - \mu_i(-))^2}{(\sigma_i(+)^2 + \sigma_i(-)^2)} \tag{2.7}$$

These selection methods eliminate noisy features but can result in redundant genes being kept, and are unable to recognise complementary features that individually do not seperate the data well.

Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a computationally efficient greedy feature selection algorithm introduced by Guyon et al.. It recursively removes

features with the smallest absolute weights in a hyperplane produced by an SVM.

The RFE algorithm is:

1. Train an SVM to distinguish between classes using all remaining features.
2. Calculate the ranking criterion, w_i^2, for each feature. Where w_i^2 is the square of the hyperplane component that feature i contributes.
3. Remove the feature with lowest ranking criterion.
4. Continue to 1 unless only the desired number of genes remain.

RFE allows more than one feature to be removed per iteration, although this may lead to a subset of features which is not optimal. Unlike correlation coefficients, RFE recognises gene interactions and can remove redundant genes (Guyon et al. 2002). RFE does require an SVM to be trained for every iteration though.

2.3.5 One Vs. All

The problem of combining binary classifiers has been studied in the computer science literature (Allwein 2000) (Guruswami & Sahai 1999) from a theoretical and empirical perspective. However, the literature is inconclusive, and the best method for combining binary classifiers for any particular problem is open (Ramaswamy 2001).

To allow binary classifiers to handle multiple classes there are two major approaches, One Verses All (OVA) and All Pairs (AP). Using the One vs. All methodology it is possible to combine multiple binary classifiers into a multi-class classifier. A separate binary classifier is trained for each class by treating exemplars belonging to the class as positive exemplars and all others as negative. When classifying a new example the winning class is the one with the largest margin. How this margin is created is dependant on the classification method and can also be thought of as a signed confidence measure.

For N classes we have N binary OVA classifiers, $(f_1 \dots f_N)$, each of which output a margin value. The class is then determined by:

$$class \quad = \quad \texttt{max}_{i=1\dots N}(f_i) \tag{2.8}$$

For any new instance being classified through the OVA approach we can assign a confidence measure to its classification. This is determined by the difference from the margin, f_i, of the selected class to the next highest margin of another class.

SVM

For SVMs the margin value of a classification has direct meaning. This margin is the distance of a sample to the seperating hyperplane between the two classes.

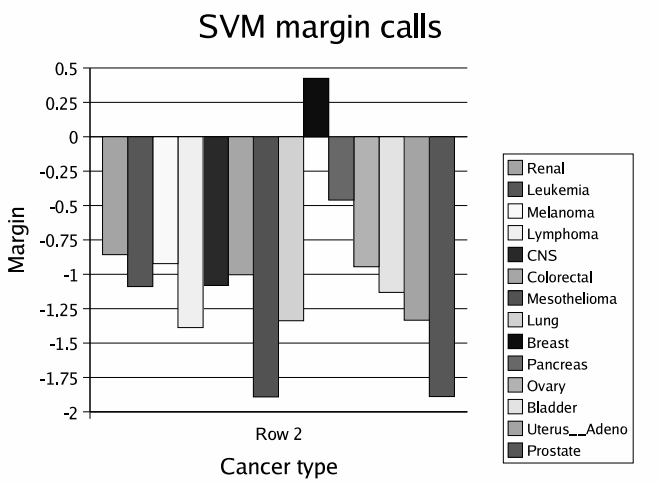


Figure 2.5: The margins of a 14 binary classifiers that together form an OVA classifier from this report. These margins are for a Breast cancer sample.

An example of an OVA SVM classifier is shown in figure 2.5. This figure shows the Breast classifier strongly matching an example, and due to the large difference between it and the next highest margin it is match of strong *confidence*.

Genetic Algorithms

The output from the Novamente genetic programming classifier is a value in the range $[0 \ldots 1]$. Where 1 reflects a strong match and 0 a rejection. This value can be taken directly and used as the margin.

Chapter 3

Technical Details

3.1 Dataset

The dataset consisted of 218 tumor samples, spanning 14 common tumor classes (see A.2). Each sample had 16063 gene expression values associated with it.

Ramaswamy divided 144 of these samples into a training set and 54 into a test set (including 8 which were from metastasic samples). The rest were poorly differentiated and added to a seperate test set. The same division of tumour samples is used in this study, however the poorly differentiated test set was not used.

See (Ramaswamy 2001) for a detailed methodology of how the microarrays used in this study were constructed, including the sources of the tissue samples.

3.1.1 Preprocessing

Preprocessing operations are those that occur to a dataset prior to more detailed analysis in order to make it more suited to the analysis methods or research question. The three preprocessing operations often used in application to microarray data are (Holland 1975):

- **Thresholds** - Thresholds are upper and lower numerical limits applied to outlier values. Negative expression values are thought to be unreliable and cannot easily be understood in physiological terms (Mutch et al. 2001). To overcome these negative values low expression values are rounded to an arbitary lower limit of between 20 and 100 expression units.
- **Filters** - filters remove certain genes from a dataset. This can help to eliminate noise and reduce the size of the dataset, as well as inhibit overfitting (see section 2.3.4). Filters are also used to limit genes based on some form of biological criterion, such as which chromosome they lie on.
- **Transformations** - transformations globally change a dataset with mathematical operations. An example is the logarithmic transform, which is

especially useful when applied to microarray data because variance has been observed to be proportional to signal intensity. Gene expression values at high signal intensities are less reproducable than lower intensities (Nadon & Shoemaker 2002)

All the gene expression values in the dataset used in this report had a threshold of 20 to 16000 applied, before being logarithmicly transformed. Each feature (gene) was then linearly mapped to the range [0..1] with the minimum at 0 and the maximum at 1.

3.2 Implementations

3.2.1 Support Vector Machine

A modified version of SVMTorch (Collobert & Bengio 2001) was used to generate SVM models. The modifications[1] enabled recursive feature elimination to be carried out.

(Ramaswamy 2001) showed that a linear kernel could create a hyperplane able to fully seperate the training data when all 16063 features were incorporated into the model. (Ramaswamy 2001) justified the choice of a linear kernel by the dataset consisting of relatively few data points in a large number of dimensions. Similarly in this report, a linear SVM was used.

A Lagrange multipliers limit value of 100 was specified and an epsilon value of 0.5 used for tolerance[2].

3.2.2 Feature Selection

The least significant 60% of features were removed at each iteration of RFE.

RFE was performed within the modified version SVMTorch for SVM classifiers. At the end of an RFE run in SVMTorch a summary of the features removed at each step was created. This summary was then passed to a purpose built Java application to create data files for classification by GPC[3]. Each file represented a step in the RFE process.

Each binary classifier in an OVA scheme is independant of the others, so in RFE they select the features which are most significant in that particular class versus all others. This creates a seperate hierarchy of feature sets for each binary classifier.

For the feature selection approach using a correllation coefficient, the definition by (Pavlidis et al. n.d.) was chosen:

$$w_i = \frac{(\mu_i(+) - \mu_i(-))^2}{(\sigma_i(+)^2 + \sigma_i(-)^2)} \tag{3.1}$$

[1] Partly implemented by Bernardo P. R. Carvalho of Biomind (bernardo@vettatech.com)

[2] As suggested by the SVMTorch manual.

[3] Framework code for parsing the WICGR RES file format supplied by Lúcio de Souza Coelho (lucio@vettatech.com)

Data files were created using this measure for the corellation coefficient and a similar Java application to that used for RFE data file creation. The same number of features as for each iteration of RFE (removing the least significant 60%) were selected for each file.

The correllation coefficient for each feature changes with the each binary classifier within an OVA group, because each class alters the mean and standard deviation of the positive and negative groups. Therefore, feature selection by corellation coefficient, like RFE, results in a hierarchy of feature sets for each binary classifier within an OVA scheme.

3.2.3 Genetic Program Classifier

The code for the GPC implemented within Novamente was altered to allow invididual OVA classifiers to each use their own hierarchy of features as determined by feature selection.

Each GPC was created with a population of 50 individuals, and a mutation rate of 0.005. Elitism was disabled, with new generations created from the winners of tournament selection.

The depth of trees within the population was set by the following formula:

$$depth \quad = \quad ceil(\frac{\log N}{\log 2}) + 1 \tag{3.2}$$

3.3 Platform

All experiments were conducted in Linux on either workstations with either Athlon XP1800+ or Intel Pentium 4 (2.40GHz) CPUs, and 512Mb RAM.

3.4 Testing Pipeline

The following steps outline the procedure for the experiments:

- Preprocess raw microarray data and save into SVMTorch format.
- Run SVMTorch with RFE for each OVA classifier saving summaries and classifying test data at each iteration.
- Preprocess raw microarray data again, but take into account RFE summary and save into Novamente format.
- Evolve GPCs within Novamente, on each iteration of RFE data.
- Preprocess raw microarray data again, calculating corellation coefficients to create feature subsets. Save subsets into Novamente format.
- Evolve GPCs within Novamente, on each iteration of corellation coeffiecient selected data.

Chapter 4

Experimental Results

4.1 Hypothesis

Two hypothesis were investigated to test the feature selection with GPCs.

Hypothesis 1 *RFE feature selection will give higher accuracy classification in GPCs.*

Hypothesis 2 *Feature selection will reduce convergence time of GPCs.*

4.2 Accuracy of Support Vector Machines

This section reports the results of SVM accuracy with feature selection and compares them with the original study by (Ramaswamy 2001). While not directly related to the hypotheses, the results from these experiments questions specific details of the original study's methodology.

Each OVA SVM classifier (14 individual binary classifiers) took an amount of time proportional to the number of features being used. In general however, they took from 10-30 minutes each to create.

Figure 4.1 shows the effect of RFE on SVM classification accuracy when the data is used raw without preprocessing and when preprocessing is used. The accuracy of SVMs applied to preprocessed data is generally better than on raw data.

Comparing the curves to the graph presented by (Ramaswamy 2001) (Figure 4.2) it suggests that the original study did not use data preprocessing. Their report states that preprocessing was used but was ambiguous as to whether it was specifically used with RFE.

The effect of sub-optimal feature selection due to removing large sets of features every iteration can be seen between Figures 4.2 and 4.1. The results of this study show much lower accuracy than Ramaswamy at low amounts of features. This is likely due to Ramaswamy removing the least significant 10%

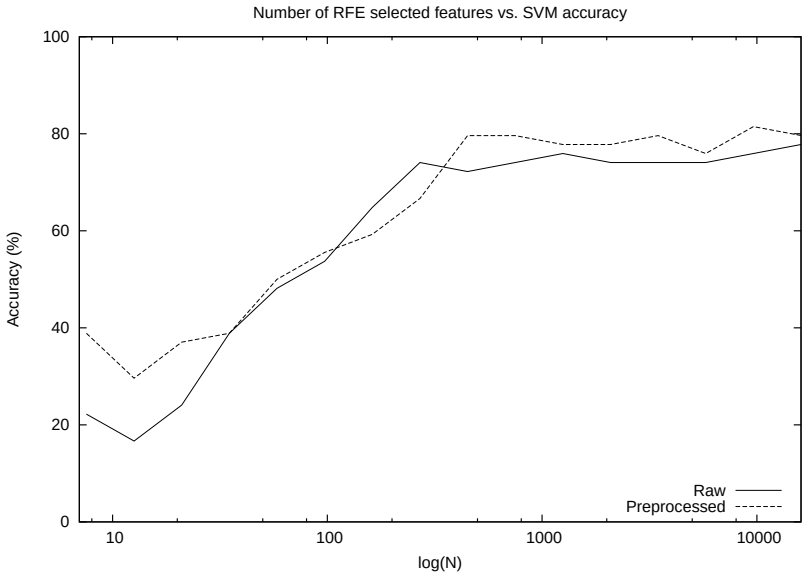


Figure 4.1: The effect on classification accuracy with increasing numbers of features (N)

of features rather than the 40% removed in this study. Removing more features at every iteration, results in less optimal subsets being created.

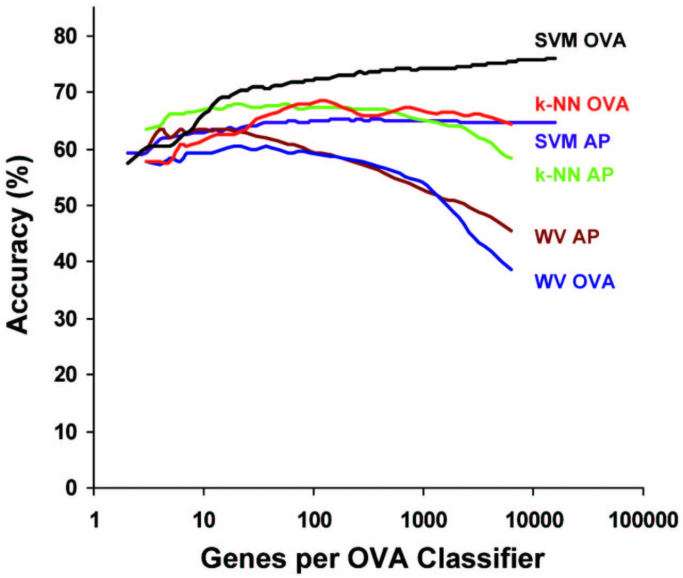


Figure 4.2: Classification accuracy of SVMs in study by Ramaswamy, also shown are results for k Nearest Neighbour and Weighted Voting classifiers. Graph from `http://www.pnas.org/cgi/content/full/98/26/15149`

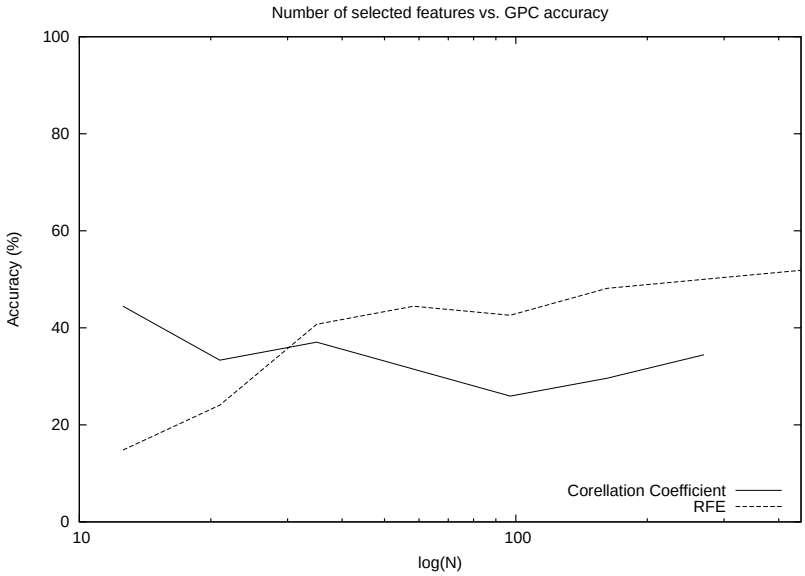


Figure 4.3: The effect on GPC classification accuracy with increasing numbers of features (N)

4.3 Genetic Program Classifiers

Here we examine the the validity of the two hypotheses from Section 4.1.

4.3.1 Accuracy

The feature selection method that results in the best GPC accuracy depends on the number of features present at a particular iteration, as shown in Figure 4.3. At low numbers of features (< 20), the corellation coefficient selection method results in the best accuracy. Whereas higher numbers of features (> 20) give better accuracy when selected with RFE.

This is possibly due to depth of tree being limited at lower feature numbers preventing complex rules evolving which are needed for the combined significance of RFE selected features to emerge.

Thus, Hypothesis 1 is correct for more than twenty features present, but for less than this, the corellation coefficient feature selection method is best. This validation is made with the reservation that shallow trees at a low number of features may be preventing the significance of RFE selected genes being utilised.

4.3.2 Convergence

To investigate the convergence of GPCs, the accuracy of individual binary classifiers within an OVA group were analysed, as well as the average fitness for

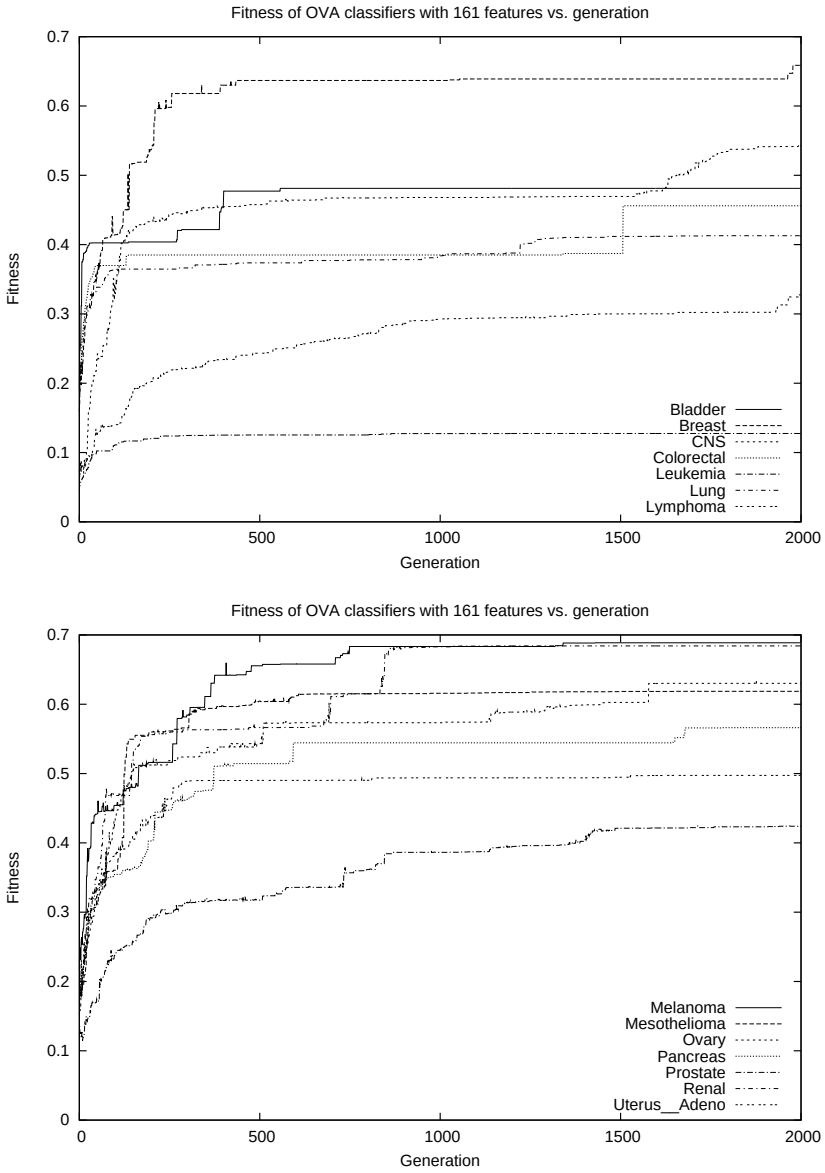


Figure 4.4: Fitness of OVA classifiers with 161 features (RFE selected) vs. generation (N)

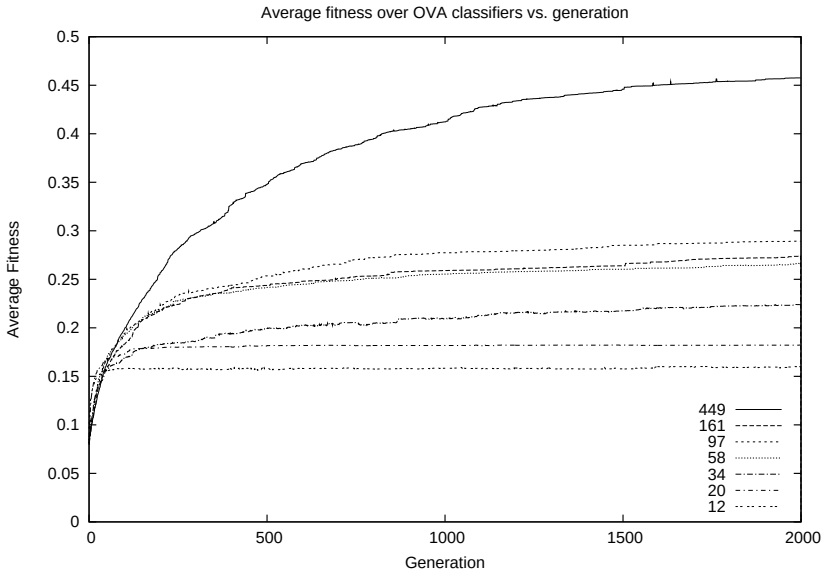


Figure 4.5: Average fitness across OVA classifiers (features RFE selected) vs. generation (N)

OVA groups.

Between binary classifiers in an OVA group we often observe large jumps in fitness (see Figure 4.4. These probably represent a significant gene, along with its relation to the class, being included in the GPC function trees. For particular classes, markedly sharp increases in fitness occured. These "jumps" in fitness may indicate a particular gene being found that is heavily involved in the development of that cancer type. The most marked jumps were observed in Colorectal, and also to a lesser extent in Bladder.

There is variation in fitness levels for the various GPC binary classifiers (and consequently their ability to distinguish between cancer types), which may give an idea of the relative difficulty of distinguishing cancer classes. For example, Leukemia and Lymphoma both have relative low fitness, whereas Renal cancer and Melanoma have high fitnesses. This may indicate that that Renal and Melanoma have distinct patterns that classifiers can pick up, while the patterns for Leukemia and Lymphoma may be more subtle.

Another observed phenomenom is that some classes reach a "plataeu", or converge, faster than others. The Leukemia, Uterus, and Bladder classes show this behaviour.

The sharp peaks that occur in the binary clasifier fitnesses indicate that the most fit individual was not involved in tournament selection and hence did not continue to the next generation.

Figure 4.5 presents the fitness of an OVA classifier based on the average fitness across all its binary classifiers. Fast convergence is observed for a low

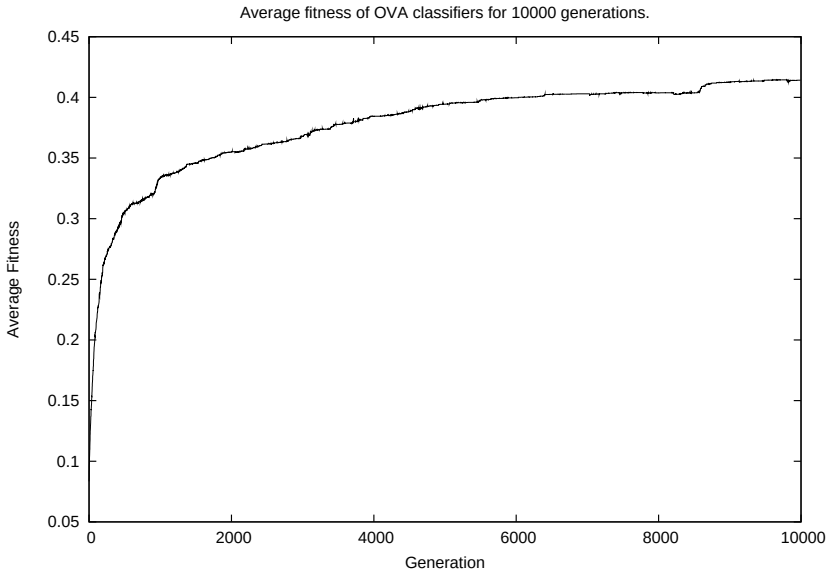


Figure 4.6: Average fitness across OVA classifiers for 10000 generations with 161 features.

number of features, and this supports hypothesis 2.

However, this may be a result of varying the tree depth of GPCs. More complex rules can be created by deeper trees, but this complexity will take longer to emerge through evolutionary selection. Convergence to an optimal value would be quicker for shallow trees.

The results presented so far don't indicate whether an ideal solution is reached. To investigate the timing required to search the solution space for an ideal classifier, a OVA GPC was run for 10000 generations with a eventual accuracy of 51.9% on the test set. The rate of fitness increase slows (see Figure 4.6) and seems to approach an asymtope which presumably represents the ideal solution. The fitness does not reach a constant value in the number of generations observed. There is also a sharp increase just before 9000 generations, indicating that fundamental changes are still being made late in the process, therefore it hasn't reached optimal solution despite taking over 60 hours to execute.

4.4 Selected Features

Table 4.1 displays the five most significant RFE selected genes, based on their contributions to the SVM hyperplane.

It should be noted that many of the genes appear more than once. Specifically hum_alu_at (and hum_alu_at-2) and L06499_at, the latter being a Ribo-

Table 4.1: The five most significant genes for each cancer class as determined by RFE.

Cancer Class	RFE selected genes	
Bladdder	L06499_at	M62895_s_at
	AA422123_f_at	D79205_at
	Z12962_at	
Breast	L06499_at	Z12962_at
	U57847_s_at	X16869_s_at
	M62895_s_at	
CNS	hum_alu_at-2	X16869_s_at
	AA422123_f_at	AFFX-HSAC07/X00351_3_at-2
	AFFX-HSAC07/X00351_3_at	
Colorectal	hum_alu_at-2	hum_alu_at
	U57847_s_at	AFFX-HSAC07/X00351_3_at-2
	AFFX-HSAC07/X00351_3_at	
Leukemia	hum_alu_at-2	hum_alu_at
	L06499_at	M62895_s_at
	U57847_s_at	
Lung	hum_alu_at-2	hum_alu_at
	L06499_at	M62895_s_at
	X57351_s_at	
Lymphoma	AFFX-HSAC07/X00351_M_at-2	AFFX-HSAC07/X00351_M_at
	AFFX-HSAC07/X00351_5_at-2	AFFX-HSAC07/X00351_5_at
	AFFX-HUMGAPDH/M33197_5_at-2	
Melanoma	hum_alu_at-2	hum_alu_at
	L06499_at	M62895_s_at
	Z12962_at	
Mesothelioma	M17885_at	X17206_at
	L06499_at	X69150_at
	HG3214-HT3391_at	
Ovary	hum_alu_at-2	hum_alu_at
	X57351_s_at	M62895_s_at
	AA422123_f_at	
Pancreas	L06499_at	X17206_at
	M17885_at	U14973_at
	AFFX-HSAC07/X00351_M_at-2	
Prostate	hum_alu_at-2	hum_alu_at
	L06499_at	M62895_s_at
	U57847_s_at	
Renal	V00594_s_at	hum_alu_at
	hum_alu_at-2	AFFX-HSAC07/X00351_3_at-2
	AFFX-HSAC07/X00351_3_at	
Uterus_Adeno	hum_alu_at-2	hum_alu_at
	L06499_at	M62895_s_at
	U57847_s_at	

Table 4.2: Description of the genes found significant by RFE.

Gene name	Description
L06499_at	RPL37A Ribosomal Protein L37A
M62895_s_at	Annexin II (lipocortin II) pseudogene 2
AA422123_f_at	zv26h12.r1 Soares NhHMPu S1 Homo sapiens cDNA clone 754823 5' similar to contains Alu repetitive element; mRNA sequence (from GenBank)
D79205_at	Ribosomal protein L39
Z12962_at	EEF1A1 Translation eolongation factor 1-alpha-1
U57847_s_at	Ribosomal protein S27 (metallopanstimulin 1)
X16869_s_at	Eukaryotic translation elongation factor 1-alpha-1
AFFX-HSAC07/X00351_3_at-2	none
AFFX-HSAC07/X00351_3_at	
hum_alu_at-2	
hum_alu_at	
X57351_s_at	RPS3 Ribosomal protein S3
AFFX-HSAC07/X00351_M_at-2	none
AFFX-HSAC07/X00351_M_at	
AFFX-HSAC07/X00351_5_at-2	none
AFFX-HSAC07/X00351_5_at	
AFFX-HUMGAPDH/M33197_5_at-2	Glyceraldehyde-3-phosphate dehydrogenase
M17885_at	RPLP0 Ribosomal protein, large, P0
X17206_at	Ribosomal protein L26
X69150_at	Ribosomal protein S18
HG3214-HT3391_at	Metallopanstimulin 1
U14973_at	Ribosomal Protein S29
V00594_s_at	Metallothionein isoform 2

somal protein. Indeed, it seems that a majority of the selected genes have an association with the Ribosome (Table 4.2). The Ribosome is a cellular organelle predominantly involved in protein synthesis from mRNA. This process is called translation, and two genes (X16869_s_at and Z19262_at) involved in translation are also present in our set of RFE selected genes.

Of particular interest are the four sequences of unknown function belonging which were determined to significant in determining Lymphoma. These could be further researched in an attempt to discover more about their function in relation with Lymphoma.

Chapter 5

Discussion

The results for convergence are likely to be influence by the depth of trees used in creating the GPCs. The varying tree depths are necessary to allow the GPC to take into consideration all of the features if it needs to. To investigate this, the experiments should be repeated with a constant tree depth. This depth should be large enough to accomodate all the features of the largest feature subset used.

The time taken for training GPCs is much longer than SVMs. This is due to two factors, the number of generations and tree depth, both of which need to be large in order to get useful classification. The time taken will be partly influenced by Novamente using processor time to maintain its system and perform MindAgent activities. Although SVMs are much quicker, GPCs result in a decision tree that Novamente can absorb.

It was also realised that the absence of evolutionary *elitism* was likely to be slowing the speed at which GPC fitness increases, since fit solutions can be lost through random picking of individuals to compete in tournament selection.

5.1 Further Work

Alot more work presents itself in the domain of machine learning applied to classification of gene expression data. This is reflected by the large number of related articles currently being published in the literature.

Areas of particular interest to this study follow.

5.1.1 Rejection calls

As a natural consequence of using an OVA methodology, a measure of confidence for any particular classification can be made.

This is done by finding the difference between the two highest margin values of the OVA binary classifiers. This difference can be thought of the confidence of classification.

Table 5.1: Rercursive Feature Selection with exhaustive search and model selection (RFEX) compared to plain Recursive Feature Selection (RFE)

	RFEX		RFE	
Class	# Features	Accuracy	# Features	Accuracy
Leukemia	2	100%	64	98%
Colon	4	98%	4	86%

If machine learning methods are used as a diagnosis tool for oncologists, then it would be useful to have a threshold of confidence. Where any classification with a confidence less than this threshold is rejected, the oncologist could then either redo the gene expression sample or fall back to traditional methods of diagnosis.

5.1.2 Alternatives to GPC

Although the use of genetic algorithms to classify cancer based on gene expression is particularly poignant, GPCs are slow. Since the motivation for using GPCs was to create a classifier in a form Novamente can absorb and use with the rest of its system, other methods of creating classifiers may be investigated.

Rule induction methods generalise the training set into rules that can be used to classify new examples, which make them ideal alternatives to GPCs. In particular, ID3 (Quinlan 1986) is a rule-based method that reduces a set of training examples to a decision tree.

The decision tree of ID3 is produced in a top-down fashion. Beginning at the root node, attributes are chosen to discriminate between classes with each value of the attribute producing a subnode. ID3 continues choosing attributes to discriminate on at each node until all the examples that node have the same class. A commercial version of ID3 called C4.5 (Quinlan 1993) supports continuous value domains (which are produced by microarrays), by partitioning the range of possible values. C4.5 also includes tree pruning which prevents overfitting and could thus perform a similar function to feature selection.

5.1.3 Feature Selection Methods

There are many methods of selecting features from within a dataset, and these may result in selected features that are better for classification by GPCs.

One such method that is based on RFE combined with an exhaustive search of feature combinations along with model selection (Guyon et al. 2001). They report that it eliminates gene redundancy, and yields better and more complex gene subsets. Table 5.1. shows the number of genes selected by this method and their accuracy in classification of cancer types based on gene expression.

5.1.4 Gene Clusters

Clustering is a form of unsupervised learning that where it is assumed the classes of samples are initially unknown. This technique could be useful for grouping similar genes, and for discovering sub-groups within a cancer class.

One method of clustering is the Self Organasing Map (SOM), where one randomly chooses the geometry of a grid (e.g., a 2 x 4 grid) and maps it into the N-dimensional feature space. Initially the features are randomly mapped to the grid but during training the mapping is iteratively adjusted to better reflect the structure of the data (Eisen et al. 1998).

5.1.5 Regulatory Network Inference

Microarrays give a glimpse of the expression levels of thousands of genes, and can be considered as the state of a system or network as overviewed by (Pitt 2003). As long as we have a time series of expression levels it is possible to try and infer a network of how the genes involved interact. Some may promote other genes, while others inhibit them. This is known as an "inverse problem" where the dynamics of a system are trying to be elucidated from data (Crutchfield 1989).

Regulatory network inference (RNI) entails working out which genes repress one another and how they can cascade and cause chain reactions. For example, if gene A is being expressed, what effect is that likely to have on gene B's expression in the future? Often these networks are incredibly complex, especially when they involve thousands of genes.

The Biomind LLC (Limited Liability Corporation), whom use Novamente as a tool to analyse biological data, have to some extent already investigated using Novamente for RNI (Biomind 2003). It has not been compared to other methods of RNI yet, and this is primarily due to other methods not being tested on real data. In order to test how accurate a method is, one has to know the underlying network to compare to the one the method generates. Currently no real data of such genetic regulatory networks exist, and so networks are randomly generated and simulated to create output similar to that of a microarray.

It should be noted that RNI is an extension of trying to predicting values based on the past behaviour of a gene's expression. As any network that is inferred should have prediction power, as well as a host of other characteristics as outlined by (Wessels et al. 2001).

Chapter 6

Conclusion

This study has investigated using feature selection on gene expression data in order to make the use of Genetic Programming Classifiers (GPCs) feasible. The results presented suggest that feature selection using coefficient corellation ranking is best for reducing the number of features to small number (< 20) of features. For greater numbers of features Recursive Feature Elimination via the use of Support Vector Machines is preferential as it gives higher accuracy.

The accuracy given by GPCs is not sufficient for practical use in the diagnosis of patients, but may still suggest relationships amoung genes that may interesting to study further. An alternative method for creating structures for Novamente to reason may be sought by using ID3 to create decision trees.

The effect of feature selection on the convergence time was also examined. The conclusion being drawn that a smaller number features means a plateau of fitness values is reached faster than with a large number of features.

Both of the conclusions of this report are dependant on the depth of GPC function trees increasing with the number of features. Since it is likely that this on its own would also give the observed results, it is suggested that further investigation be pursued.

Appendix A

Dataset

The dataset used in this report is available for public download from `http://www-genome.wi.mit.edu/MPR/GCM.html`. A list of samples within the training and test datasets is displayed in Tables A.1 and A.2.

A.1 Training Samples

Class	Sample Name	Pathology
Breast	mBRT1 (8697)	Adenocarcinoma, Invasive Lobular
	mBRT2 (9078)	Adenocarcinoma, Invasive Lobular
	95 I 029	Adenocarcinoma
	94 I 155	Adenocarcinoma, Moderately to Poorly Differentiated
	92 I 078	Adenocarcinoma
	9912c068 CC	Infiltrating ductal, Moderately Differentiated
	93 I 250	Mucinous Adenocarcinoma, Moderately to Poorly Differentiated
	94 I 159	Mucinous Adenocarcinoma
Prostate	94 I 052	Adenocarcinoma, High Grade
	95 I 249	Adenocarcinoma, High Grade
	LocalCaP1T	Adenocarcinoma
	LocalCaP10T	Adenocarcinoma
	P 0025	Adenocarcinoma
	P 0030	Adenocarcinoma
	P 0033	Adenocarcinoma
	P 0036	Adenocarcinoma
Lung	004 B	Adenocarcinoma
	009 C	Adenocarcinoma
	93 I 226	Adenocarcinoma
	93 I 146	Adenocarcinoma
	HCTN LUT1 (18702 A1F)	Adenocarcinoma, Poorly Differentiated
	HCTN LUT4 (18870 A1C)	Adenocarcinoma, Moderately to Poorly Differentiated

	94 I 196	Adenocarcinoma
	H 20387	Adenocarcinoma, Moderately to Poorly Differentiated
Colorectal	mCRT1 (8936)	Adenocarcinoma, Moderately Differentiated
	mCRT2 (9752)	Adenocarcinoma
	95 I 057	Adenocarcinoma
	HCTN CRT1 (18851 A1B)	Adenocarcinoma, Moderately Differentiated
	0001c038 CC	Adenocarcinoma, Moderately to Poorly Differentiated
	0001c040 CC	Adenocarcinoma
	9912c055 CC	Adenocarcinoma, Moderately to Poorly Differentiated
	HCTN 19339	Adenocarcinoma, Well Differentiated
Lymphoma	L_B_CELL_S93_20626_Y	Lymphoma, Large B-cell
	L_B_CELL_S98_8825_Y	Lymphoma, Large B-cell
	L_B_CELL_S93_04233_Y	Lymphoma, Large B-cell
	L_B_CELL_S94_17150_R	Lymphoma, Large B-cell
	L_B_CELL_S98_12569_R	Lymphoma, Large B-cell
	L_B_CELL_S98_12453_R	Lymphoma, Large B-cell
	L_B_CELL_S98_17557_R	Lymphoma, Large B-cell
	L_B_CELL_S94_22323_G	Lymphoma, Large B-cell
	FSCC_S98_11020	Lymphoma, Follicular
	FSCC_S98_10416	Lymphoma, Follicular
	FSCC_S98_5894	Lymphoma, Follicular
	FSCC_S98_14359	Lymphoma, Follicular
	FSCC_S93_13188	Lymphoma, Follicular
	FSCC_S93_20082	Lymphoma, Follicular
	FSCC_S93_14386	Lymphoma, Follicular
	FSCC_S93_23356	Lymphoma, Follicular
Bladder	11520	Transitional Cell Carcinoma, Granular and Squamous Differentiation
	9858	Transitional Cell Carcinoma, Poorly Differentiated
	94 I 229	Transitional Cell Carcinoma
	07-B 003E	Transitional Cell Carcinoma
	B_0001	Transitional Cell Carcinoma, Papillary
	B_0002	Transitional Cell Carcinoma
	B_0003	Transitional Cell Carcinoma, Papillary, Poorly Differentiated
	B_0004	Transitional Cell Carcinoma
Melanoma	94 I 149	Melanoma, Poorly Differentiated
	94 I 191	Melanoma
	0-9652	Melanoma
	93 I 262	Melanoma
	96 I 166	Melanoma
	11337	Melanoma
	MGH 10427	Melanoma
	MGH 11511	Melanoma
Uterus	92 I 073	Adenocarcinoma, Moderately to Poorly Differentiated
	4203	Adenocarcinoma

	3663	Adenocarcinoma
	2967	Adenocarcinoma
	5116	Adenocarcinoma
	3226	Adenocarcinoma
	4915	Adenocarcinoma
	2552	Adenocarcinoma
Leukemia	AML (1,PK) BM	Leukemia, Acute Myelogenous
	AML (2,PK) BM	Leukemia, Acute Myelogenous
	AML (3,PK) BM	Leukemia, Acute Myelogenous
	AML (5,PK) BM	Leukemia, Acute Myelogenous
	AML (6,PK) BM	Leukemia, Acute Myelogenous
	AML (7,PK) BM	Leukemia, Acute Myelogenous
	AML (12,PK) BM	Leukemia, Acute Myelogenous
	AML (13,PK) BM	Leukemia, Acute Myelogenous
	ALL (16415) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (19881) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (9186) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (9723) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (17269) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL(14402) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (17638) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (22474) BM	Leukemia, Acute Lymphocyitc, T-cell
	ALL (19769) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (23953) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (28373) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (9335) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (9692) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (14749) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (17281) BM	Leukemia, Acute Lymphocyitc, B-cell
	ALL (19183) BM	Leukemia, Acute Lymphocyitc, B-cell
Renal	92 I 126	Renal Cell Carcinoma, Moderately to Poorly Differentiated
	Carc_609TO	Renal Cell Carcinoma
	Carc_628TG	Renal Cell Carcinoma
	Carc_614TS	Renal Cell Carcinoma
	Carc_614TO	Renal Cell Carcinoma, Moderately to Poorly Differentiated
	Carc_623TS	Renal Cell Carcinoma
	Carc_623TO	Renal Cell Carcinoma
	5291	Renal Cell Carcinoma
Pancreas	95 I 298 (I)	Adenocarcinoma, Moderately Differentiated
	Pan 1T	Adenocarcinoma, Poorly Differentiated
	Pan 2T	Adenocarcinoma
	Pan 3T	Adenocarcinoma
	Pan 4T	Adenocarcinoma
	Pan 6T	Adenocarcinoma
	Pan 7T	Adenocarcinoma

	Pan 16T	Adenocarcinoma
Ovary	mOVT1 (8691)	Adenocarcinoma, Mixed Serous and Transitional
	mOVT2 (I) (9334)	Adenocarcinoma, Serous Papillary, Poorly Differentiated
	93 I 081	Adenocarcinoma
	HCTN 19155	Adenocarcinoma, Endometroid
	HCTN 0002c011N	Adenocarcinoma, Serous Papillary
	07-B 001B	Adenocarcinoma, Papillary, Poorly Differentiated
	07-B 014G	Adenocarcinoma, Serous Papillary
	H (400)6346	Adenocarcinoma, Clear Cell
Mesothelioma	161 T6	Mesothelioma, Pleural
	31 T10	Mesothelioma, Pleural
	169 T7	Mesothelioma, Pleural
	165 T5	Mesothelioma, Pleural
	228 T4	Mesothelioma, Pleural
	235 T6	Mesothelioma, Pleural
	74 T6	Mesothelioma, Pleural
	215 T5	Mesothelioma, Pleural
CNS	GlioB_1	Glioblastoma
	GlioB_2	Glioblastoma
	GlioB_3	Glioblastoma
	GlioB_4	Glioblastoma
	GlioB_5	Glioblastoma
	GlioB_6	Glioblastoma
	GlioB_7	Glioblastoma
	GlioB_8	Glioblastoma
	M29_A91	Medulloblastoma
	M22_A50	Medulloblastoma
	M42_A10	Medulloblastoma
	M20_A56	Medulloblastoma
	M49_A2	Medulloblastoma
	M06_D92	Medulloblastoma
	M02_D16	Medulloblastoma
	M50_D7	Medulloblastoma

A.2 Test Samples

Class	Sample Name	Pathology
Breast	93 I 192	Adenocarcinoma
	09-B 005A	Adenocarcinoma, Infiltrating Ductal, Moderately Differentiated
	09-B 003A	Adenocarcinoma, Invasive Ductal
Prostate	95 I 256	Adenocarcinoma
	P 0062	Adenocarcinoma, Poorly Differentiated
Lung	H 20154	Adenocarcinoma, Moderately Differentiated
	LT14	Adenocarcinoma

	95 I 117	Adenocarcinoma, Moderately to Poorly Differentiated
Colorectal	HCTN 19389	Adenocarcinoma
	0001c026 CC	Adenocarcinoma
	95 I 175	Adenocarcinoma, Moderately Differentiated
Lymphoma	L_B_CELL_S94_6696_G	Lymphoma, Large B-cell
	L_B_CELL_S97_27534_G	Lymphoma, Large B-cell
	L_B_CELL_S98_1217_R	Lymphoma, Large B-cell
	FSCC_S93_11021	Lymphoma, Follicular
	FSCC_S99_9100	Lymphoma, Follicular
	FSCC_S99_5073_1	Lymphoma, Follicular
Bladder	B_0007	Transitional Cell Carcinoma, Moderately to Poorly Differentiated
	B_0008	Transitional Cell Carcinoma
	104-64931	Transitional Cell Carcinoma
Melanoma	MGH 8907	Melanoma
	MGH 7974	Melanoma
Uterus	4075	Adenocarcinoma
	4840	Adenocarcinoma
Leukemia	AML (14,PK) BM	Leukemia, Acute Myelogenous
	AML (16,PK) BM	Leukemia, Acute Myelogenous
	ALL (92_571) BM	Leukemia, Acute Lymphocytic, T-cell
	ALL (87_52) BM	Leukemia, Acute Lymphocytic, T-cell
	ALL (20414) BM	Leukemia, Acute Lymphocytic, B-cell
	ALL (21302) BM	Leukemia, Acute Lymphocytic, B-cell
Renal	6775	Renal Cell Carcinoma
	5382	Renal Cell Carcinoma
	6727	Renal Cell Carcinoma
Pancreas	Pan 29T	Adenocarcinoma
	Pan 17T	Adenocarcinoma
	97 I 077	Adenocarcinoma
Ovary	0001c086	Adenocarcinoma, Papillary Serous, Poorly Differentiated
	H 6206	Adenocarcinoma, Endometrioid
	HCTN 19120	Adenocarcinoma, Papillary Serous, Poorly Differentiated
Mesothelioma	166 T4	Mesothelioma, Pleural
	224 T5	Mesothelioma, Pleural
	300 T	Mesothelioma, Pleural
CNS	GlioB_9	Glioblastoma
	GlioB_10	Glioblastoma
	M51_D34	Medulloblastoma
	M33	Medulloblastoma
Ovary	9912c062cc_Rb	Metastases to Peritoneum, Adenocarcinoma
Colorectal	HCTN 19274	Metastases to Liver, Adenocarcinoma, Moderately Differentiated
Lung	H 20300	Metastases to Liver, Adenocarcinoma
Prostate	MetCaP1	Metastases to Bone, Adenocarcinoma
Prostate	MetCaP109	Metastases to Bone, Adenocarcinoma
Prostate	MetCaP125	Metastases to Bone, Adenocarcinoma

Prostate	MetCaP128	Metastases to Bone, Adenocarcinoma
Breast	MGH_4934	Metastases to Lymph Node

Bibliography

Allwein (2000), 'Reducing multiclass to binary: A unifying approach for margin classifiers', *Proc ICML 2000* .

Baker, J. (1985), Adaptive selection methods for genetic algorithms, *in* 'Proceedings of an International Conference on Genetic Algorithms and their Application', Hillsdale, New Jersey, p. 101111.

Biomind (2003), 'Regulatory pattern inference and algorithmic technique'.

Brown, M., Grundy, W., Lin, D., Christianini, N., Sugnet, C., Furey, T., Ares, M. & Haussler, D. (2000), *Proc Natl Acad Sci USA* **97**, 262–267.

Collobert, R. & Bengio, S. (2001), 'Support vector machines for large-scale regression problems', *J Machine Learning Res* **1**, 143–160.

Connolly, J. L., Schnitt, S. J., Wang, H. H., Dvorak, A. M. & Dvorak, H. F. (1997), Williams & Wilkins, Baltimore, pp. 533–555.

Cristianini, N. & Shawe-Taylor, J. (1999), *An introduction to support vector machines*, Cambridge University Press.

Crutchfield, J. (1989), Inferring the dynamic, quantifying physical complexity, *in* A. Albano, N. Abraham, P. Rapp & A. Passamonte, eds, 'Measures of Complexity and Chaos', Plenum Press.

Darwin, C. (1859), *The Origin of Species by means of Natural Selection or the Preservation of Favoured Races in the Struggle for Life.*

Dawkins, R. (1996), *Climbing Mount Improbable*, Penguin.

Duda, R. O. & Hart, P. E. (1973), *Pattern classification and scene analysis*, Wiley.

Eisen, M. B., Spellman, P. T., Brown, P. O. & Botstein, D. (1998), 'Cluster analysis and display of genome-wide expression patterns', *Proc. Natl. Acad. Sci. USA* **95**, 1486314868.

Evgeniou, T., Pontil, M. & Poggio, T. (2000), *Advances in Computational Mathematics* **13**, 1–50.

Fodor, S. A. (1997), 'Massively parallel genomics', *Science* **277**, 393–395.

Furey, T., Cristianini, N., Duffy, N., Bednarski, D., Schummer, M. & Haussler, D. (2000), 'Support vector machine classification and validation of cancer tissue samples using microarray expression data', *Bioinformatics* **16(10)**, 906–14.

Goertzel et al., B. (2003+), *Novamente: A design for a digital mind*. Second draft.

Goldberg, D. (1989), *Genetic algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.

Golub, T. R., Slonim, D. K., Tamayo, P., C., H., Gaasenbeek, M. & Mesirov, J. P. (1999), 'Molecular classification of cancer: class discovery and class prediction by gene expression monitoring', *Science* **286(5439)**.

Guruswami, V. & Sahai, A. (1999), Multi-class learning, boosting and error-correcting codes., *in* 'Proceedings of the Twelfth Annual Conference on Computational Learning Theory', ACM Press, pp. 145–155.

Guyon, I., Weston, J., Barnhill, S. & Vapnik, V. (2001), Gene selection for cancer classification using support vector machines, Paper, Barnhill Bioinformatics and AT&T Labs.

Guyon, I., Weston, J., Barnhill, S. & Vapnik, V. (2002), 'Gene selection for cancer classification using support vector machines', *Machine Learning* **46(1-3)**.

Holland, J. (1975), *Adaption in natural and artificial systems*, The University of Michigan Press.

Kauffman, S. (1995), *At Home in the Universe: The search for the Laws of Complexity*, Penguin, London.

Kearns, M., Mansour, Y., Ng, A. & Ron, D. (1997), 'An experimental and theoretical comparison of model selection methods', *Machine Learning* **27**, 750.

Kohavi, R. & John, G. (n.d.), 'Wrappers for feature subset selection', *Artificial Intelligence* **97**, 273–324.

Kuravilla, F. G., Park, P. J. & L., S. S. (2002), 'Vector algebra in the analysis of genome-wide expression data', *Genome Biol* **3(3)**.

Michalewicz, Z. (1996), *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd edn, Springer Verlag.

Minsky, M. & Papert, S. (1972), *Perceptrons: An introduction to computational geometry.*, MIT Press.

Mukherjee, S. (1999), 'Support vector machine classification of microarray data.', *CBCL Paper* **182**.
*http://www.ai.mit.edu/projects/cbcl/publications/ps/cancer.ps

Mutch, D. M., Berger, A., Mansourian, R., Rytz, A. & Roberts, M. A. (2001), 'Microarray data analysis: a practical approach for selecting differentially expressed genes', *Genome Biol* **2(12)**.

Nadon, R. & Shoemaker, J. (2002), 'Statistical issues with microarrays: processing and analysis', *Trends Genet* **18(5)**.

Nilsson, N. J. (1996), Unpublished.
*http://robotics.stanford.edu/people/nilsson/mlbook.html

Pavlidis, P., Weston, J., Cai, J. & Grundy, W. N. (n.d.), 'Gene functional analysis from heterogeneous data', *Submitted for publication* .

Pitt, J. (2003), 'Inferring genetic regulatory networks from microarray gene expression data: a systems perspective.'.

Quinlan, J. (1986), 'Induction of decision trees', *Machine Learning* **1**, 81–106.

Quinlan, J. (1993), *C4.5; Program for Machine Learning*, Morgan Kaufman.

Radmacher, M. D., McShane, L. M. & Simon, R. (2002), 'A paradigm for class prediction using gene expression profiles', *J Comput Biol* **9(3)**.

Ramaswamy, S. (2001), 'Multiclass cancer diagnosis using tumor gene expression signatures.', *Proc Natl Acad Sci USA.* **98(26)**, 15149–54.

Ramaswamy, S., Osteen, R. T. & Shulman, L. N. (2001), Clinical oncology, pp. 711–719.

Rosenblatt (1962), *Principles of Neurodynamics*, Spartan Books, New York, NY.

Tomaszewski, J. E. & LiVolsi, V. A. (1999), *Cancer* **86**, 2198–2200.

Vapnik, V. N. (1998), *Statistical Learning Theory*, John Wiley & Sons, New York.

Wessels, L., van Someran, E. & Reinders, M. (2001), 'Genetic network models: a comparative study', *Proceedings of SPIE, Micro-arrays: Optical Technologies and Informatics (BIOS01)* **4266**, 236–247.