11-27-2007

Improving Feature Selection Techniques for Machine Learning

Feng Tan

Recommended Citation

Tan, Feng, "Improving Feature Selection Techniques for Machine Learning." Dissertation, Georgia State University, 2007.
doi: https://doi.org/10.57709/1059437

IMPROVING FEATURE SELECTION TECHNIQUES FOR MACHINE LEARNING

by

FENG TAN

Under the Direction of Anu G. Bourgeois

ABSTRACT

As a commonly used technique in data preprocessing for machine learning, feature selection identifies important features and removes irrelevant, redundant or noise features to reduce the dimensionality of feature space. It improves efficiency, accuracy and comprehensibility of the models built by learning algorithms. Feature selection techniques have been widely employed in a variety of applications, such as genomic analysis, information retrieval, and text categorization.

Researchers have introduced many feature selection algorithms with different selection criteria. However, it has been discovered that no single criterion is best for all applications. We proposed a hybrid feature selection framework called based on genetic algorithms (GAs) that employs a target learning algorithm to evaluate features, a wrapper method. We call it hybrid genetic feature selection (HGFS) framework. The advantages of this approach include the ability to accommodate multiple feature selection criteria and find small subsets of features that perform well for the target algorithm. The experiments on genomic data demonstrate that ours is a robust

and effective approach that can find subsets of features with higher classification accuracy and/or smaller size compared to each individual feature selection algorithm.

A common characteristic of text categorization tasks is multi-label classification with a great number of features, which makes wrapper methods time-consuming and impractical. We proposed a simple filter (non-wrapper) approach called Relation Strength and Frequency Variance (RSFV) measure. The basic idea is that informative features are those that are highly correlated with the class and distribute most differently among all classes. The approach is compared with two well-known feature selection methods in the experiments on two standard text corpora. The experiments show that RSFV generate equal or better performance than the others in many cases.

INDEX WORDS: Feature selection, Gene selection, Term selection, Dimension Reduction, Genetic algorithm, Text categorization, Text classification

IMPROVING FEATURE SELECTION TECHNIQUES FOR MACHINE LEARNING

by

FENG TAN

A Dissertation Submitted in Partial Fulfillment of Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia Stage University

2007

IMPROVING FEATURE SELECTION TECHNIQUES FOR MACHINE LEARNING

by

FENG TAN

Committee Chair:	Anu G. Bourgeois
Committee:	Yanqing Zhang
	Robert Harrison
	Yichuan Zhao

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

December 2007

ACKNOWLEDGEMENTS

Firstly, my special thanks go to my supervisor, Dr. Anu G. Bourgeois. She always gives me the freedom to define and explore my own research direction. At the mean time, she provides her prompt help and guidance whenever I encounter problems. The dissertation would not have been possible without her help.

Secondly, I would like to thank my committee members, Dr. Yan-Qing Zhang, Dr. Robert Harrison and Dr. Yichuan Zhao for their well-appreciated support and assistance.

Finally, I want to thank my family and friends for their support and beliefs. They have encouraged me and provided unconditional support for my Ph. D study through these years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

CFS	Correlation-based Feature Selection
CV	Cross Validation
DF	Document Frequency
GA	Genetic Algorithm
IG	Information Gain
IR	Information Retrieval
KNN	K-nearest Neighbor
LOO	Leave One Out
MI	Mutual Information
NB	Naïve Bayes
OR	Odds Ratio
RFE	Recursive Feature Elimination
RSFV	Relation Strength and Frequency Variance
SBS	Sequential Backward Search
SFS	Sequential Forward Search
SVM	Support Vector Machine
SU	Symmetrical Uncertainty
TC	Text Categorization
TF	Term Frequency
tfidf	Term Frequency and Inverse Document Frequency

Chapter 1

Introduction

Recent technological developments such as the Internet, database, hyperspectral imagery, and DNA microarray have facilitated the emergence of vast amounts of multivariate data in a wide spectrum of applications including search engines, genomic analysis, proteomics, image retrieval, information retrieval, and text categorization. Unfortunately, the growth of data volume far outpaces human's ability to manage and understand them. Machine learning provides tools to alleviate the problem by automatically analyzing large quantities of data. However, applications with hundreds to thousands of features (attributes) make it challenging for machine learning to extract useful information from gigantic data streams.

Feature selection that selects a subset of most salient features and removes irrelevant, redundant and noisy features is a process commonly employed in machine learning to solve the high dimensionality problem. It focuses learning algorithms on most useful aspects of data, thereby making learning task faster and more accurate. In this dissertation, we are interested in improving feature selection techniques for machine learning.

1.1 Motivation

Machine learning is a field that studies algorithms and techniques that allow computer programs to automatically improve with experiences, that is, to "learn". Learning algorithms are provided with data that exemplify a task so that they can learn from and predict the new data. A

decade ago, few domains explored data with more than 40 features. This situation has changed considerably in the past few years, due to the emergence of new application domains [$27]. Tasks with a large size of feature space present a new challenge for existing learning algorithms [$51, $15, $4, $68].

Gene selection and text categorization problems are two examples typical of the new application domains with high dimensional data. In gene selection problems [$102, $56, $58], expression levels of many genes are recorded by microarray data, but only a small number of discriminatory genes are critical for cancer classification and diagnosis. In addition, compared to the large size of features (i.e. genes), usually small size of examples (e.g. fewer than 100) are available altogether for training and testing, which makes learning even more difficult. In text categorization problems [$36, $107, $1, $20], feature space is determined by the vocabularies from the natural language documents whose size is commonly of hundreds of thousands of words. Meanwhile the collection of documents available for classification is typically large. For instance, numerous Web pages and online articles are available, but we are only interested in searching for those of a particular topic, which is a very small fraction of the whole.

Feature selection that studies how to select informative (or discriminative) features and remove irrelevant, redundant or noisy ones from data, is an important and frequently used technique for data preprocessing in machine learning. By reducing the dimensionality of data, feature selection reduces the overall computational cost, improves the performance of learning algorithms and enhances the comprehensibility of the data models. With the help of feature selection, machine learning algorithms become more scalable, reliable and accurate.

Many feature selection algorithms have been proposed in the literature [$57, $55, $103, $52, $16, $102, $27, $5, $106, $54]. One group called wrapper employs learning algorithms to

evaluate features, while the other group called filter is independent of any learning algorithm by using intrinsic properties of the data to assess features. Since feature selection criteria proposed are very diverse and motivated by various theoretic arguments, they often produce substantially different outcomes when even applied to same data set. It has been noted that various selection criteria are biased with respect to dimensionality and no single criterion is best for all applications [$21, $9]. This discordance caused by various selection criteria makes the interpretation of the data difficult. Moreover, it causes difficulty in determining which feature selection method best suits new data. Hence, we believe exploring ways to combine multiple criteria or to develop multi-objective criteria seems a reasonable approach to study.

We proposed a hybrid genetic feature selection (HGFS) framework, also a wrapper method, for feature selection. HGFS framework is based on the genetic algorithms (GAs) that combines various feature selection criteria. The goal is to effectively utilize useful information from different feature selection methods to select better feature subsets with smaller size and/or higher classification performance than the individual feature selection algorithms. Another advantage of the method is that it can find feature subsets that are best suited for a target learning algorithm. The framework is applied to several gene selection applications. The experiments on microarray data show that the proposed hybrid feature selection method is capable of achieving the goal [$94, $96, $95].

Text categorization applications generally have massive data samples and features, which makes wrapper methods rather time-consuming and impractical for these applications. For this reason, the use of faster and simpler filter approaches is prominent in the domain [$87, $108, $22, $101, $80, $12, $69]. We proposed a simple filter approach named Relation Strength and Frequency Variance (RSFV) measure. It is based on the principle that important features should

be highly correlated with the class and distribute most differently among all the classes. We conducted experiments on two standard text corpora and compared RSFV with two widely used feature selection methods. The experiments demonstrate that our approach obtains comparable or better results than the others in many situations [$93].

1.2 Organization

The remainder of the disseration is organized as follows:

Chapter 2 first introduces the background in machine learning. Then, it gives an overview of a typical feature selection procedure that consists of subset generation, subset evaluation, stopping criterion and result validation. We also review previous work in the field of feature selection.

In Chapter 3, we propose our hybrid genetic feature selection (HGFS) framework, which includes several components, a feature pool, a genetic algorithm, and an induction algorithm. The framework combines multiple feature selection criteria through a genetic algorithm. We also briefly discuss three existing feature selection methods (entropy-cased, T-statistics, and SVM-RFE) and Support Vector Machine (SVM) that are used in later experiments.

We applied our framework on two gene selection applications, which select critical genes for cancer diagnosis. Chapter 4 presents the experimental results of our approach on two DNA microarray datasets and compares them with the three existing methods covered in Chapter 3. This work has been presented at [$94, $95].

In Chapter 5, we improve our framework by using a different genetic algorithm. The new genetic algorithm is designed to achieve a balance between two goals: maximum accuracy and

minimum feature size. The framework with improved genetic algorithm is applied to the same microarray datasets described in Chapter 3. This work was presented at [$96, $95].

Chapter 6 introduces the area of text categorization, where we are interested in applying feature selection techniques as well. Definitions and concepts in text categorization area are described. We briefly explain a typical text categorization process, in which feature selection is an important step.

In Chapter 7, we propose a simple feature selection metric called Relation Strength and Frequency variance (RSFV) measure for text categorization. RSFV evaluates features according to their correlations with the classes and their distributions among the classes. We compared RSFV with two existing feature selection method in experiments. This work is submitted to SIAM International Conference on Data Mining (SDM08) [$93].

Chapter 8 concludes our work and gives several future directions.

Chapter 2

Feature Selection for Machine Learning

With the rapid development of novel technologies and applications, larger and more complex data accumulate at an unprecedented speed. Because relevant features are often unknown, large quantities of candidate features are collected to represent these data. Although irrelevant features do not affect the target concept learnt through machine learning and redundant features do not add anything new to the target concept in any way [$37], they drastically increase the computation cost of a learning process. In many real-world tasks, the dimentionality of data is so high that it is computationally costly or practically prohibitive for machine learning. Many traditional learning algorithms fail to scale on large-size problems due to the curse of dimensionality. In addition, the existence of noisy features degrade the performance of learning algorithms. Feature selection techniques are developed to solve these problems in machine learning [$53, $15, $4].

2.1 Introduction to Machine Learning

The field of machine learning is concerned with the study of algorithms and techniques that allow computers to automatically "learn" from experiences. Machine learning draws on concepts and techniques from many fields, including statistics, information theory, artificial intelligence, biology, philosophy, and cognitive science. In general, there are two types of learning: inductive and deductive. Inductive machine learning algorithms generalize or extract knowledge (i.e. rules

and patterns) that are unknown before out of data examples. On the other hand, deductive learning works on existing knowledge and deduces new knowledge from the old.

Supervised Machine Learning

In supervised learning, the class labels of training data are already known. The training examples are represented as pairs of an input object and its desired output (e.g., class label). The task of a supervised learner is to find a function to approximate the mapping between training data and their classes so that it can predict the classes of new data. There are many approaches and algorithms proposed for supervised learning, such as artificial neural networks [$46], naive Bayes classifiers [$45], decision trees [$75], K-nearest neighbor [$13], support vector machines (SVMs) [$8] and random forests [$5].

Unsupervised Machine Learning

Unsupervised learning is distinguished from supervised learning by the fact that the class labels of training data are not available. Unsupervised learning methods decide which objects should be grouped together as one class. In other words, they learn classes by themselves. K-nearest neighbor [$13], self-organizing maps (SOMs) [$42], and data clustering algorithms [$33] (e.g., K-means clustering, fuzzy c-means clustering) are often used for unsupervised learning tasks.

A good representation of input objects is important because the accuracy of the learned model depends strongly on how the input object is represented. Typically, the input object is transformed into a vector of features or attributes that are used to describe the object.

2.2 Feature Selection

Feature selection (also known as subset selection or variable selection) is a process commonly employed in machine learning to solve the high dimentionality problem. It selects a subset of important features and removes irrelevant, redundant and noisy features for simpler and more concise data representation. The benefits of feature selection are multi-fold. First, feature selection greatly saves the running time of a learning process by removing irrelevant and redundant features. Second, without the interference of irrelevant, redundant and noisy features, learning algorithms can focus on most important aspects of data and build simpler but more accurate data models. Therefore, the classification performance is improved. Third, feature selection can help us build a simpler and more general model and get a better insight into the underlying concept of the task [$41, $43, $15].

Feature selection is different from feature extraction (or feature transformation), which creates new features by combining the original features. Principal component analysis (PCA) [$38], linear discriminant analysis (LDA) [$59], and locally linear embedding (LLE) [$81] are examples of feature transformation techniques. On the other hand, feature selection maintains the original meanings of the selected features, which is desirable in many domains.

2.3 Feature Selection Objectives

Different feature selection algorithms may have various objectives to achieve. The following is a list of common objectives used by researchers [$15]:

1. Find the minimally sized feature subset that is necessary and sufficient to the target concept [$40].
2. Select a subset of *N* features from a set of *M* features, $N < M$, such that the value of a criterion function is optimized over all subsets of size *N* [$66].
3. Choose a subset of features for improving prediction accuracy or decreasing the size of the structure without significantly decreasing prediction accuracy of the classifier built using only the selected features [$43].
4. Select a small subset such that the resulting class distribution, given only the values for the selected features, is as close as possible to the original class distribution given all feature values [$43].

2.4 Feature Selection Procedure

A typical feature selection procedure (shown in Figure 2.1) consists of four basic steps: 1) subset generation; 2) subset evaluation; 3) stopping criterion and 4) result validation [$53]. The process begins with subset generation that employs a certain search strategy to produce candidate feature subsets. Then each candidate subset is evaluated according to a certain evaluation criterion and compared with the previous best one. If it is better, then it replaces the previous best. The process of subset generation and evaluation is repeated until a given stopping criterion is satisfied. Finally the selected best feature subset is validated by prior knowledge or some test data. Search strategy and evaluation criterion are two key topics in the study of feature selection.

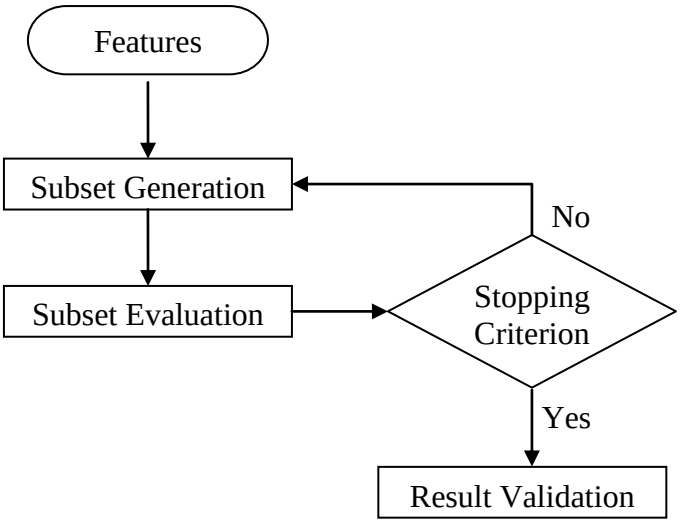


Figure 2.1 Feature selection procedure

2.5 Subset Generation

Subset generation begins with a search start point, which can be an empty set, the full set, or a randomly generated subset. From the start point, it can search feature subsets from different directions, such as forward, backward, and random. In forward search, features are added one at a time, while in backward search the least important feature is removed based on evaluation criterion. Random search adds or deletes features at random to avoid being trapped into a local maxima.

There are various search strategies for finding an optimal or suboptimal feature subset. As we know, if the full feature set contains N features, the total number of candidate subsets is 2^N. An exhaustive search strategy searches all 2^N feature subsets to find an optimal one. Its complexity is exponential (i.e. $O(2^N)$) in terms of the data dimension. When applied to high dimensional data

sets, finding an optimal feature subset is usually intractable [$41]. Therefore, many heuristic search strategies have been developed to circumvent this problem.

The branch and bound method proposed by Narendra and Fukunaga [$66] basically performs an exhaustive search in an orderly fashion (e.g. a search tree), but stops the search along a particular branch if some limit or bound is exceeded, or if the sub-solution does not look very promising. It guarantees to find an optimal feature subset. In spite of its time complexity of $O(2^N)$, the branch and bound method is fast for certain problems.

Some algorithms use greedy hill climbing strategies [$82], a simple local search that chooses the change that maximally decreases the cost of the solution. Once a change is accepted, it is never backtracked. Sequential forward search (SFS), sequential backward search (SBS), and bidirectional search [$51] are some variations to the greedy hill climbing method. For these methods, the local change is simply the addition or deletion of features from the subset. Just as its name implies, SFS sequentially searches the feature space. It starts from the empty set and selects the best single feature to add into the set in each iteration. Just the opposite, SBS starts from the full feature set and removes the worst single feature from the set in each iteration. Both approaches add or remove features one at a time. Algorithms with sequential searches are fast in time complexity of $O(N^2)$ and simple to implement. It is often argued that forward selection is computationally more efficient than backward elimination to generate nested subsets of variables [$27]. However, the defenders of backward elimination argue that weaker subsets are found by forward selection because the importance of variables is not assessed in the context of other variables not included yet [$27].

Best first search [$78] is similar to greedy hill climbing as it searches the feature space by making local changes to the current feature subset. However, unlike hill climbing, it allows

backtracking if the path being explored seems unpromising. It will return to a previous promising point to continue searching from there.

Other feature selection algorithms randomly search for the solutions, such as evolutionary algorithms [$106, $34, $98, $44] and simulated annealing [$18]. The use of randomness helps to avoid local optima in the search space.

2.6 Evaluation Criteria

After feature subsets are generated, they are evaluated by a certain criterion to measure their goodness. Generally, the goodness of feature subsets means the discriminating ability of subsets to distinguish among different classes. Based on whether they are dependent on the inductive learning algorithms, feature selection algorithms can be broadly divided into three categories: wrapper, filter, and hybrid.

2.6.1 Wrapper Method

In a wrapper method, the performance (e.g. classification or prediction accuracy) of an induction algorithm of interest is used for feature subset evaluation. Figure 2.2 show the ideas behind wrapper approaches [$41]. For each generated feature subset *S,* wrappers evaluate its goodness by applying the induction algorithm to the dataset using features in subset *S*. Wrappers can find feature subsets with high accuracy because the features match well with the learning algorithms.

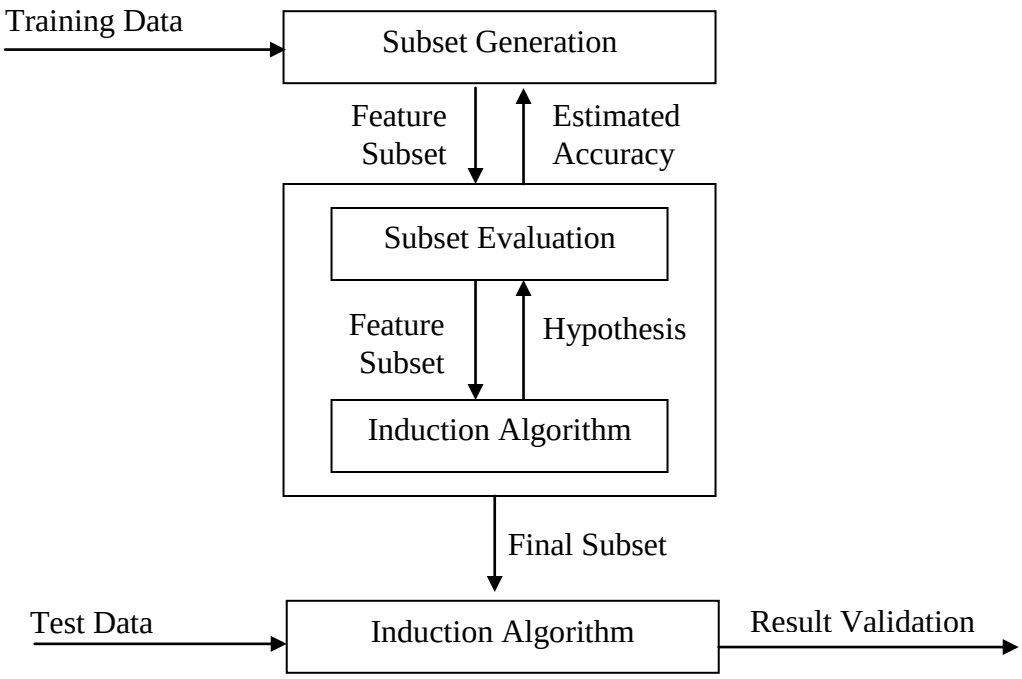


Figure 2.2 The wrapper approach for feature selection

2.6.2 Filter Method

Independently of any induction algorithm, filter methods filter out irrelevant, redundant or noisy features in preprocessing steps before induction occurs. Unlike wrappers, filters (shown in Figure 2.3) [$17, $28, $110] utilize the intrinsic properties of data to evaluate feature subsets. In general, features are assessed by their relevance or discriminatory powers with regard to target classes.

The general argument of wrapper approaches is that the induction method that measures feature subsets should provide a better estimate of accuracy than a separate measure that may have entirely different inductive bias [$4]. That is, wrappers generally provide better performance in terms of classification accuracy than filters.

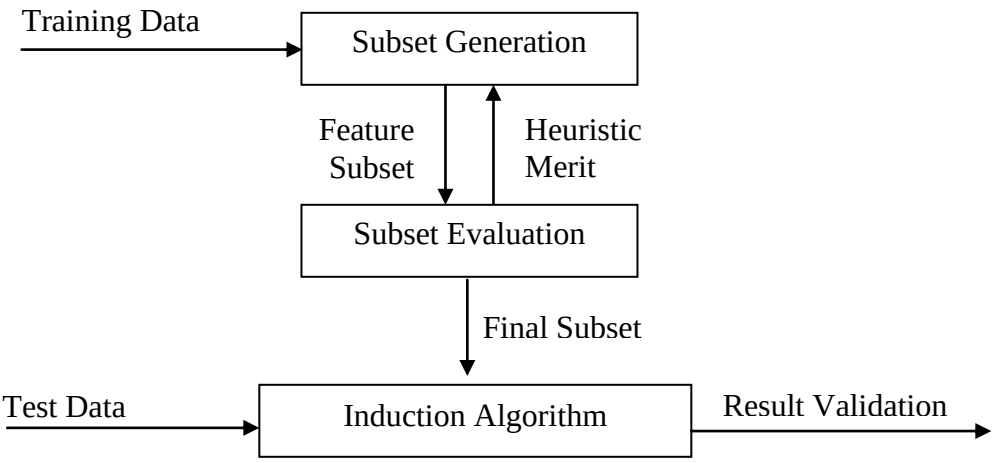


Figure 2.3 The filter approach for feature selection

However, wrappers typically require more extensive computation than filters, which results from repeatedly calling the induction algorithm to evaluate each candidate feature subset. Furthermore, since the subset evaluation is tightly coupled with a learning algorithm, wrappers must be re-run when a different learning algorithm is to be used for feature evaluation. It is argued that filters have better generalization properties than wrappers because they are independent of any specific learning method [$54].

2.6.3 Hybrid Method

To improve classification performance and fasten feature selection, one can build hybrid models [$14, $105] that take advantage of filters and wrappers by using both independent criteria and learning algorithms to measure feature subsets. Filters can provide an intelligent guideline for wrappers, such as a reduced search space, a good starting point, or a smarter/shorter search path, which help scale wrappers to larger size problems. Typically, a hybrid method uses the

independent measure to decide the best subsets for a given cardinality and uses the learning algorithm to select the final best subset among the best subsets across different cardinalities [$53]. It usually starts with an initial empty subset and iterates to find the best subsets in the order of increasing cardinality.

2.6.4 Feature Ranking

Among the proposed feature selection algorithms, feature ranking approaches that score or rank features by certain criterion and use rankings of features as the base of selection mechanism are particularly attractive because of their simplicity, scalability, and good empirical success [$27]. Computationally, feature ranking is efficient since it requires only the computation of M scores and sorting the scores. Based on the ranks of features, subsets of significant features can be selected to build a predictor or classifier.

Different researchers have introduced varying feature selection criteria. Some filter methods use ranking criteria based on statistics, such as χ^2-statistics [$52], T-statistics [$55], F-statistics [71$], MIT correlation (also known as signal-to-noise statistic) [$26], and Fisher criterion [24$]. Some use information-theoretic criteria including information gain [$57], mutual information [27$, 71$], and entropy-based measure [16$, 56$, 17$]. Other wrapper approaches utilize machine learning algorithms, such as Support Vector Machines (SVMs) [102$, 103$, 58$], and decision trees [5$] for feature ranking and selection.

Hsu et al. [32$] studied the behavior and relationship between rank combination and score combination by introducing a concept called rank/score graph. They showed that under certain conditions, rank combination outperforms score combination. Chuang et al. [9$] applied rank

combination to combine different feature selection methods. The ranks of features are combined by using a weighted sum (or average) from each of the component rankings obtained from individual feature selection method. It is showed that the combination approach performs better than each individual feature selection method in many cases. Other researchers [80$, 69$] also reported that a further increase in performance was obtained by combining various feature selectors. The combinations are simply done by using the maximum, minimum, or average of ranks or normalized scores.

2.7 Stopping Criteria

A feature selection process can be terminated under one of the following criteria [15$]:

1. Whether the search is complete.
2. Whether a predefined size of feature subsets is selected.
3. Whether a predefined number of iterations are executed.
4. Whether an optimal or sufficiently good feature subset according to the evaluation function has been obtained.
5. Whether the change (addition or deletion of features) of feature subsets does not produce a better subset.

2.8 Result Validation

In some applications, the relevant features are known beforehand. Then we can validate the feature selection results by this prior knowledge. However, in most real-world applications we

do not know which features are relevant. We have to use the classification performance on test data as an indicator of the goodness of the selected feature subsets.

Chapter 3

A Hybrid Genetic Feature Selection Framework

As mentioned in Chapter 2, wrappers generally give better results in terms of the quality measure of a learning algorithm than filters because they find feature subsets that are optimized for the learning algorithm used. However, the results may lose generality because the feature selection depends on a particular learning algorithm. On the other hand, filters do not inherit biases of any learning methods and they are more computationally efficient than wrappers.

Some researchers [21$, 9$] have pointed out the problem that employing diverse feature selection criteria (either using independent evaluation criteria or using induction algorithms) often produce substantially different outcomes. This is because criteria based on different theoretic arguments introduce various biases toward some aspects. For instance, in wrapper methods, using different learning algorithms to evaluate features can produce different outcomes for this reason. Consequently, the performance of the classifiers built upon these feature selection methods varies as well. The problem leads to a dilemma: the more algorithms available, the more challenging it is to choose a suitable one for a particular application [53$]. A good understanding of the application domain and the technical details of the available algorithms are needed to make the right choice, which is impractical in most situations. For new unknown data, it will be even more difficult to choose an appropriate method. Therefore, in this dissertation, we propose a hybrid genetic feature selection (HGFS) framework, which is based on genetic algorithms that suits different applications by combining multiple feature selection criteria [94$].

3.1 The Architecture

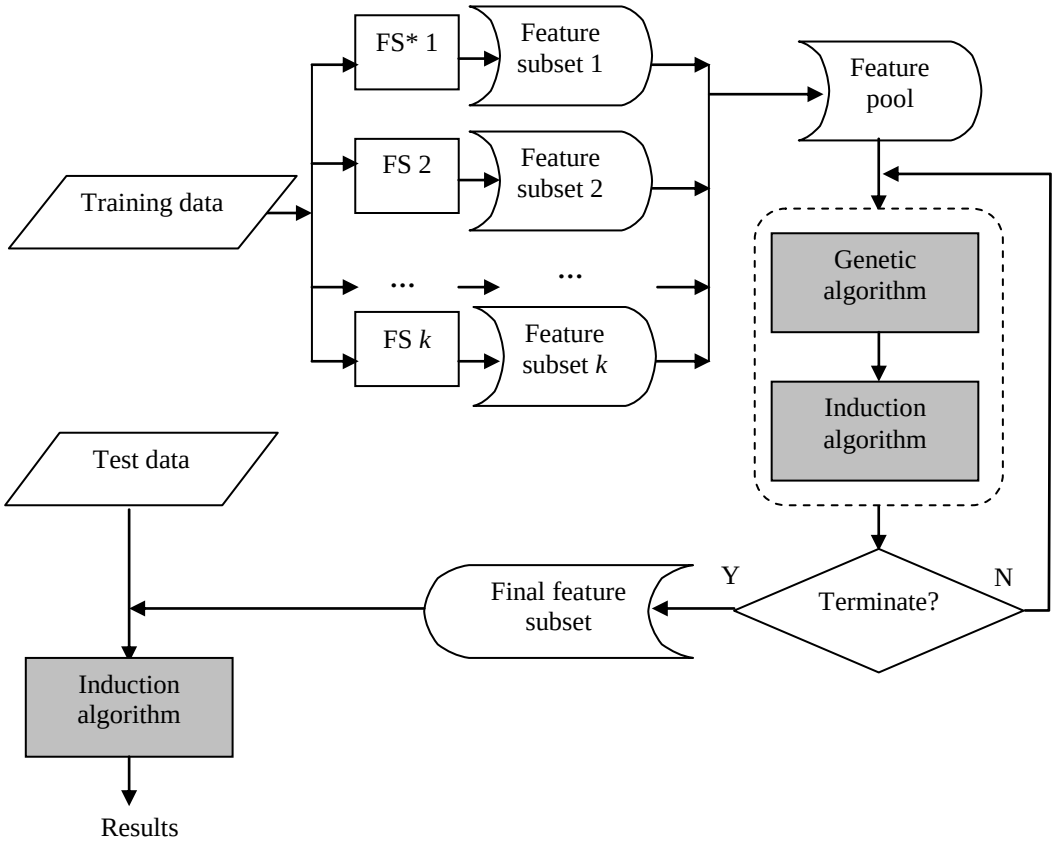
The basis behind our framework is that although different feature selection approaches often select various feature subsets, all of them provide meaningful insights into the features of application data. By extracting valuable outcomes from multiple feature selection algorithms, we are capable of finding better subsets of informative features in terms of smaller size and/or classification performance than the individual algorithms. Moreover, due to the fusion of multiple feature selection criteria, the framework is robust against various applications.

Figure 3.1 shows the architecture of our hybrid feature selection framework. In the first stage, a feature pool is formed by the inputs from a few feature selection methods. That is, several existing feature selection methods are first applied to the data and their outputs (i.e. the feature subsets selected) are fed into the feature pool as inputs to the GA. In the second stage, the genetic algorithm combines multiple feature selection criteria and searches feature subsets from the feature pool. The framework is a wrapper method based on genetic algorithms that use inductive learning algorithms to evaluate the goodness of feature subsets. In the third stage, the selected feature subset is validated by test data.

3.1.1 Feature Pool

The feature pool is a collection of candidate features to be selected by the genetic algorithm to find an optimal or near optimal feature subset. Instead of using all original features from the data, we take features selected by multiple feature selection algorithms to form the pool. In addition, the feature pool can include the features that are selected by human experts. Thus, the

feature pool contains valuable outcomes from different selection criteria and provides a good starting point for the search. In other words, we rule out some unimportant features beforehand and only consider those good features that are selected by different feature selection criteria.



FS*: Feature Selection Algorithm
Figure 3.1 The hybrid feature selection framework

Some feature selection algorithms automatically generate a subset of important features, while others produce a mere ranking of features. In the latter case, we need to determine how to select feature subsets from the ranking. A simple and common way to do it is to set a cut-off point for a ranked list of features to obtain a feature subset. However, given a ranking of features, it is unclear how to threshold the ranking to select only important variables and to exclude that

are pure noise. One common practice is to simply select the top-ranked features, -- say, top 20. A deficiency of this simple approach is that it leads to the selection of a redundant subset. Several recent studies have addressed such redundancy [71$, 109$]. Theoretically, any combination or number of feature selection algorithms can be used to generate the feature pool for input to the GA.

3.1.2 Induction Algorithm

The induction algorithm is used to create the classifier and evaluation of feature subsets. The choice of induction algorithms is independent of the genetic algorithm. Therefore, different induction algorithms, such as naïve Bayes, artificial neural network, K-nearest neighbor, and decision trees can be flexibly incorporated into our method. We choose to use SVM classifier in all the experiments due to its reported superior classification performance [8$, 67$, 24$, 58$].

Support Vector Machines

Support Vector Machines (SVMs) is a new generation learning system based on the structural risk minimization principle in statistical learning theory [8$, 100$]. SVMs attempt to learn a decision hyperplane *H* that separates the positive group from negative group with maximum margin such that the minimal distance between the hyperplane and a training example is maximal. Training data are represented as $\{(X_1, y_1), (X_2, y_2), \ldots, (X_n, y_n)\}$, where X_2 is the input vector and y_i is either 1 or −1, a constant denoting the class to which the point X_2 belongs. Separating hyperplanes takes the form

$W \cdot X - b = 0$

W is a vector that points perpendicular to the separating hyperplane. Offset parameter *b* adjusts the magin. Figure 3.2 shows an example of a two-class problem and the corresponding decision hyperplanes. Although, C_0, C_1 and *H* can separate two classes perfectly, *H* is the optimal one because it maximizes the distance between H_0 and H_1.

If the data is linearly separable, the construction of the hyperplane is always possible. Otherwise, SVMs can use kernels that nonlinearly map into a higher dimensional feature space so that a separating hyperplane can be found. We adopt linear SVM in the experiments:

$$K(x_i, x_j) = < x_i, x_j > \tag{1}$$

where x_i and x_j are two data instances in a *d*-dimensional Euclidian space.

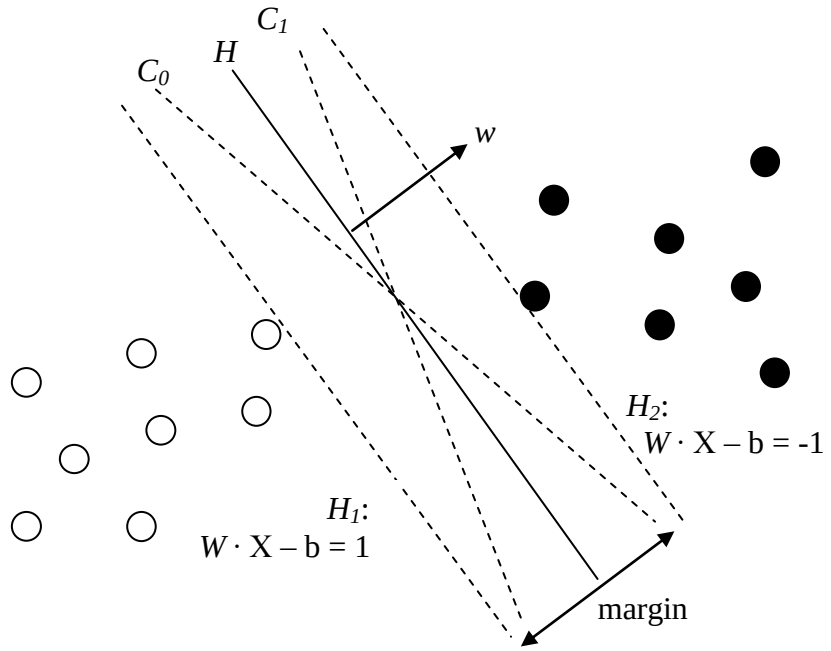


Figure 3.2 The hyperplane maximizes the margin between two classes in SVM

For a linear kernel SVM, the margin width can be calculated as the following:

$$\text{margin width} = 2/\|W\| \tag{2}$$

3.1.3 Genetic Algorithms

Genetic algorithms (GAs) provide a learning method inspired by evolutionary biology. GAs are the most popular class of evolutionary algorithms that use mechanisms such as reproduction, mutation, crossover (also called recombination), natural selection, and survival of the fittest to simulate biological evolution [31$].

Genetic algorithms have been successfully applied to a wide variety of scientific and engineering optimization or search problems. They can search spaces of hypotheses containing complex interacting parts, where the impact of each part on an overall hypothesis is difficult to model [62$]. The relative insensitivity of GAs to noise, and the requirement of no domain knowledge make them a powerful tool to optimize the process of classification, specially when the domain knowledge is costly to exploit or unavailable [98$]. Many researches demonstrate the advantages of the GAs for feature selection [106$, 6$, 76$].

Genetic algorithms begin the search for solutions in a population of initial hypotheses that traditionally are generated at random. Each hypothesis, called an individual or a chromosome, represents a potential solution of the problem. Individuals are encoded as bit strings whose interpretation depends on applications. Typically, individuals are represented in binary as strings of 0's and 1's. The initial population then evolves in generations. In each generation, every individual of the current population is evaluated according to the fitness function *F*, which is a predefined numerical measure for the problem at hand. A new population is generated by stochastically selecting the current fittest individuals. Some of the selected individuals are modified to produce new offspring individuals by mutating and recombinating parts of them. Some of these selected individuals are passed to the next generation intact. The new population

is then used in the next iteration of the algorithm. Random search strategies powered by the genetic operators (mutation and crossover) are designed to move the population away from local optima that many algorithms (e.g., greedy hill climbing) might get stuck in.

3.2 A Simple Genetic Algorithm for Feature Selection

In this section, we describe a simple genetic algorithm used in our experiments. We further developed an improved genetic algorithm for the same purpose, which is described in Chapter 5. Before we can use the genetic algorithm, there are several operations that need to be determined. They are chromosome encoding, initial population, fitness function, selection, crossover, and mutation.

3.2.1 Chromosome Encoding

We use the binary encoding scheme, a binary bit string to represent an individual. Each individual represents a candidate feature subset. The individuals are encoded by binary vectors as shown in Figure 3.3.

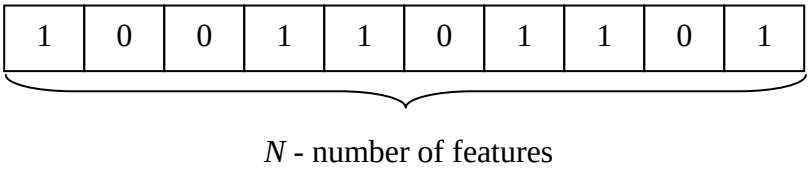


Figure 3.3: The encoding of a feature subset in the GA

Each binary digit represents a feature. The bit with value 1 means the corresponding feature is being selected, while the bit with value 0 means the opposite. The length of each chromosome is determined by the number of features *N*.

3.2.2 Initial Population

The initial population is generated randomly. For each gene *g* (bit) in the *i*th chromosome, a random function generates a random floating number within [0, 1]. If the number is over a threshold value, then $g = 1$. Otherwise, $g = 0$. For example, the threshold value can be set to 0.5 to make the chances of being 1 or 0 equal.

3.2.3 Fitness Function

The objective of the genetic algorithm here is to maximize the classification accuracy of the feature subset over the training data. Hence, the fitness of each individual $F(i)$ is defined as: $F(i) = C(i)$, where $C(i)$ is the classification accuracy when using the feature subset represented by the individual *i* to the training data.

3.2.4 Selection

Roulette wheel selection is one of the most popular selection methods for genetic algorithm. so we choose it in our genetic algorithm. Roulette wheel selection probabilistically selects

individuals from a population for later breeding. The probability of selecting individual *i* is determined by:

$$P(i) = \frac{F(i)}{\sum_{i=1}^{N} F(i)}$$

where $F(i)$ is the fitness value of *i*. The probability that an individual will be selected is proportional to its own fitness and is inversely proportional to the fitness of the other competing hypothesis in the current population. While an individual with a higher fitness will be less likely to be eliminated, there is still a chance that it may be. We used the following roulette wheel selection procedure, which is demonstrated in Figure 3.4:

1. Calculate accumulative probabilities for the *i*th individual by $p_i = \sum_{j=1,i} P(j)$ for $j = 1, \ldots, i$, where $P(j)$ is computed from the above formula.
2. Generate a random number *r* within [0, 1]
3. Select the *i*th individual if $p_{i-1} < r < p_i$

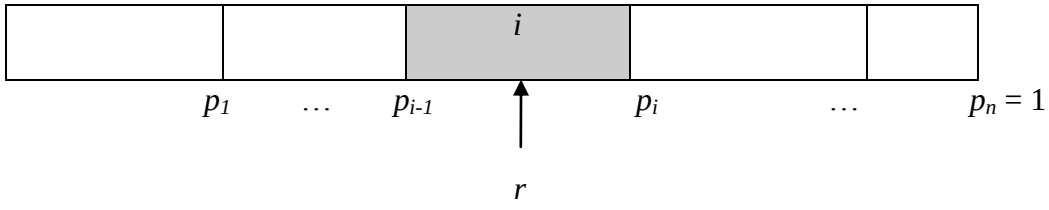


Figure 3.4 Roulette Wheel Selection

3.2.5 Crossover and Mutation

The crossover operator produces two new offsprings by copying selected bits from each parent. We use single-point crossover operator (shown in Figure 3.5). The cross-over point *i* is

chosen at random so that the first *i* bits of the two offspring are contributed by one parent and the remaining bits by the second parent.

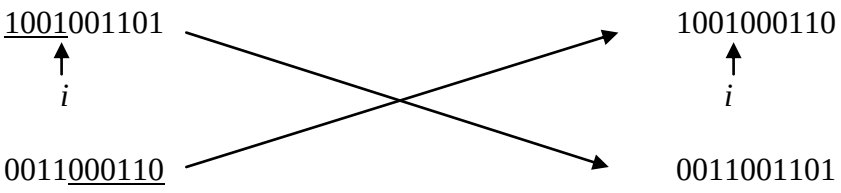


Figure 3.5 Single-point crossover

Unlike crossover, mutation produces offspring from a single parent. In particular, the mutation operator produces small random changes to the bit string by choosing a single bit at random, then changing its value. Each individual has a probability p_m to mutate. We randomly choose the *i*th position to be flipped in every mutation stage. Figure 3.6 shows how the mutation happens.

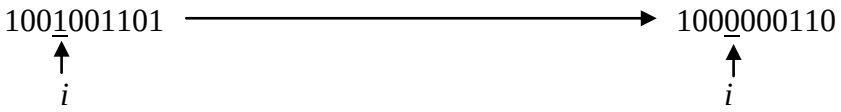


Figure 3.6 Point mutation

3.3 Feature Selection Methods

Any feature selection algorithms can be used to build the feature pool in our framework. In the experiments, we chose three existing feature selection algorithms for the feature pool, that is,

entropy-based, T-statistics, and SVM-RFE (Recursive Feature Elimination). The three algorithms are used to output candidate features into the feature pool.

Among the three feature selection algorithms we used in the experiments, there are two filters (entropy-based, T-statistics) and one wrapper (SVM-RFE). All of the three methods generate a mere ranking of features. We then pick a number of top-ranked features from each ranking and input them into the feature pool. Before presenting our experiments, we will briefly review the three feature selection algorithms used.

3.3.1 Entropy-Based Feature Ranking

The entropy-based method [16$] is based on the fact that entropy is lower for orderly configurations and higher for disorderly configurations. From this point of view, it is assumed that removing an irrelevant feature would reduce the entropy more than that for a relevant feature. The algorithm ranks the features in descending order of relevance by finding the descending order of the entropies after removing each feature one at a time. The entropy measure of a data set of N instances is calculated as the following:

$$E = -\sum_{i=1}^{N}\sum_{j=1}^{N}(S_{ij} \times \log S_{ij} + (1 - S_{ij}) \times \log(1 - S_{ij})) \quad (4)$$

$$S_{ij} = e^{-\alpha \times D_{ij}} \quad (5)$$

$$\alpha = \frac{-\ln 0.5}{\overline{D}} \quad (6)$$

where S_{ij} is the similarity measure based on distance between two instances x_i and x_j with all numeric features (similarity between two instances with nominal features is measured using Hamming distance) and α is a parameter. If one plots similarity against distance, the curve will

have a bigger curvature for a larger α. D_{ij} is the Euclidean distance between the two instances. $\overline{D}$ is the average distance among the instances. This method can be used for unsupervised data since no class information is needed.

3.3.2 T-statistics

T-statistics is a classical feature selection approach [$55] that has proven effective. It assesses whether the means of two groups are statistically different from each other. Each sample is labeled with {1, -1}. For each feature f_j, the mean μ_j^1 (resp. μ_j^{-1}) and standard deviation δ_j^1 (resp. δ_j^{-1}) are calculated using only the samples labeled 1 (resp. -1). Then a score $T(f_j)$ can be obtained by Eq. (7).

$$T(f_j) = \frac{|\mu_j^1 - \mu_j^{-1}|}{\sqrt{\frac{(\delta_j^1)^2}{n_1} + \frac{(\delta_j^{-1})^2}{n_{-1}}}} \tag{7}$$

Where n_1 (resp. n_{-1}) is the number of samples labeled as 1 (resp. -1). When making a selection, those features with the highest scores are considered as the discriminatory features.

3.3.3 SVM-RFE (Recursive Feature Elimination)

Weston et al. [102$] proposed a backward feature elimination algorithm by removing one "worst" gene (i.e., the one that changed the objective or cost function J least after being removed) at one time.

$$J = || w ||^2 / 2 \tag{8}$$

in which *w* is calculated by Eq. (2), because only linear SVM is adopted. The change of *J* caused by removing the *i*th feature is approximated by Optimal Brain Damage (OBD) algorithm [47$]:

$$\Delta J(i) = \frac{\partial J}{\partial w_i} \Delta w_i + \frac{\partial^2 J}{\partial w_i^2} (\Delta w_i)^2 \tag{9}$$

At the optimum of *J*, the first order is neglected and the second order becomes

$$\Delta J(i) = (\Delta w_i)^2 \tag{10}$$

Because removing the *i*th feature means $\Delta w_i = w_i$, w_i^2 is taken as the ranking criterion. The feature with the smallest w_i^2 is removed due to its smallest effect on classification. The iterative procedure of RFE is as follows:

1. train SVM with the training data
2. compute the ranking criterion for all features
3. remove the feature with smallest ranking criterion

In the next chapter, we will describe our use of the above three methods to generate the feature pool in the experiments.

Chapter 4

Experiments on Microarray Data

Gene selection problem is one of the typical application domains, which contains high dimensional data. In fact, it is a feature selection problem, which selects critical genes (i.e. features) from DNA microarrays for disease diagnosis. In this chapter, we apply the hybrid genetic feature selection (HGFS) framework on microarray datasets for gene selection.

4.1 Microarrays

Microarray technology is one of the recent important breakthroughs for molecular biology. It opens the possibilities of creating data sets of molecular information, which have a significant impact on molecular biology. Genes can be expressed differently at different times and under different conditions, which account for the differences in cell state or type. Microarray is a silicon chip that measures expression levels of thousands of genes in cells simultaneously. This is done by hybridizing a complex mixture of mRNAs (derived from tissue or cells) to microarrays that display probes for different genes tiled in a grid-like fashion. Hybridization events are detected using fluorescent dye and a scanner that can detect fluorescent intensities [72$]. Image analysis is performed to obtain a quantitative measure raw gene expression values. By monitoring gene expression levels, microarray leads to a more complete understanding of the molecular variations among tumors and hence to a finer and more reliable classification of

healthy or diseased cells. Many fields, including drug discovery and toxicological research, will certainly benefit from the use of DNA microarray technology.

Two practical realities constrain the analysis of microarray data [91$]. One is the "curse of dimensionality": the number of features characterizing these data is in the thousands or tens of thousands. The other is the "curse of dataset sparsity": the number of samples is comparatively limited. These two curses are believed to significantly deteriorate the performance of a classifier. Therefore, it is important to be able to remove redundant and irrelevant genes and find a subset of discriminative genes for accurate diagnosis of disease.

4.2 Experimental Setup

Cross-validation procedure is commonly used to evaluate the performance of a classifier. In *k*-fold cross-validation, the data is divided into *k* subsets of (approximately) equal size. We train the classifier *k* times, each time leaving out one of the subsets from training, but using only the omitted subset to compute the classification accuracy. Leave-one-out (LOO) cross-validation (CV) is a special case of *k*-fold cross-validation where *k* equals the sample size.

In the experiments, our focus is on using our HGFS framework to improve classification accuracy of feature subsets in comparison with each individual feature selection algorithm, not on comparing the effects of different induction algorithms (classifiers) on feature selection. We tested the framework on two microarray datasets using two different classifiers: Naive Bayes and SVM classifiers. In both cases, our feature selection approach improves the accuracy of the classifiers. Since the SVM shows superior overall classification performance than Naive Bayes,

we choose to use SVM with linear kernel as the classifier in all the experiments. However, our approach is flexible in allowing the use of different induction algorithms.

Moreover, since we are not focusing on optimizing the performance of SVMs, no efforts have been made to find the optimal parameters for SVM. In each experiment, every feature subset is classified using the same linear SVM with the same parameters. We also did not make effort to tune the parameters of the genetic algorithm used in the framework to optimize its performance. All parameters of the genetic algorithm are fixed in advance. To save time, the population size and number of generations used in the experiments by our genetic algorithm are relatively small. It is possible to achieve better results if more iterations or larger population sizes are allowed and the parameters are tuned for optimal performance.

We use two publicly available gene expression datasets (Colon Cancer and Prostate Cancer) available from [39$] in the following experiments. The datasets have already been processed in several ways, including image analysis of the microarray scanned images, dye normalization, and screening out of genes based on data quality criteria etc. All experiments are implemented on a PC with Pentium 4 (2.4GHz) and 512M RAM. All algorithms are coded in C++ and Matlab R14.

4.3 Experiment on Colon Cancer Data

The Colon Cancer dataset [39$, 2$] contains 62 tissues (samples) among which there are 40 tumor tissues and 22 normal tissues collected from colon-cancer patients. Gene expression information of colon cancer on more than 6500 genes were measured using oligonucleotide microarray and 2000 of them with highest minimum intensity were extracted to form a matrix of

62 tissues × 2000 gene expression values. For the sake of simplicity, we identify the genes (features) with their column indexes in the matrix.

Table 4.1 Top-20 Features from Entropy-Based, T-statistics, and SVM-RFE on Colon Cancer Data

Feature Selection Algorithms	Top-20 Features
Entropy-based	169, 1451, 1430, 1538, 375, 445, 1277, 1660, 603, 761, 1055, 1150, 1697, 609, 1170, 825, 1590, 1910, 803, 1264
T-Statistics	493, **1423**, 249, 377, 765, 245, 267, 66, **14**, 822, 1772, 625, 897, 137, 1674, 111, 1635, 513, 1892, 286
SVM-RFE	175, 70, **14**, 15, **1423**, 1378, 115, 164, 1791, 110, 1024, 35, 206, 38, 3, 1976, 415, 65, 16, 1325

The numbers in bold are the common gene(s)/feature(s) selected by two methods.

First, the three feature selection methods (Entropy-based, T-statistics, and SVM-RFE) are applied to the data set and three rankings of features are obtained. Due to the consideration of the cost of performing the necessary clinical test and analysis, a small sized informative gene subset (e.g. no more than 20 genes) is usually preferred for data analysis for a given accuracy. The top-20 genes ranked by the three algorithms on Colon Cancer data set are presented in Table 4.1. It shows only two genes with column index 14 and 1423 are shared by T-statistics and SVM-RFE and entropy-based has no common feature with others in top-20 features. Besides the top-20 features, we notice that the ranks of other features are also very different in the three algorithms.

Next, we pick a number of top-ranked features (e.g. top-2 features, top-4 features and so on) to get a few feature subsets. Then the SVM classifies the dataset using these feature subsets. The classification accuracies of these feature subsets presented in Table 4.2 are obtained from leave-one-out cross-validation (LOOCV) over the training data.

Table 4.2 LOO Accuracy of Entropy-based, T-statistic and SVM-RFE on Colon Cancer Data

Top Features	Entropy-based (%)	T-statistics (%)	SVM-RFE (%)
2	64.5	79.0	75.9
4	64.5	88.7	89.7
8	64.5	88.7	96.6
16	64.5	88.7	98.3
32	64.5	88.7	96.6
64	66.1	88.7	94.8
128	74.2	90.3	93.1
256	80.7	88.7	91.4
512	85.5	83.9	86.2
1024	83.9	80.7	84.5
2000	83.9	83.9	79.3

We can see that SVM-RFE provides the highest accuracy of the three except in the first case with a subset of top-2 features. With a subset of top-16 features, SVM-RFE achieves highest accuracy of 98.3% while the accuracies of the other two are 64.5% and 88.7% respectively. In general, T-statistics gives acceptable performance. Entropy-based method purely scores features based on the entropy value of the system without considering the class information, which may explain its worst classification performance of the three. However, it can be used for unsupervised data and may be less prone to overfitting. SVM-RFE assesses features by tightly binding with the classifier (SVM). It ranks features with the magnitude of the weights of a linear discriminant classifier. We think that this may account for the good performance of SVM-RFE in the experiments.

After the three rankings of features are obtained, we choose a number of top-ranked genes from the rankings and input them to the feature pool used by the GA. The classification performances of the feature selection algorithms and the domain knowledge can affect how the

feature pool is formed. The genetic algorithm uses the following parameter settings in the experiments:

- Population size: 30
- Number of generations: 10
- Probability of crossover: 1
- Probability of mutation: 0.001

Table 4.3 GA Experiments on Colon Cancer Data.

Gene/Feature Pool			Feature Subsets Selected by GA	LOO Accuracy (%)
Entropy-based	T-statistics	SVM-RFE		
Top 2 (**169,** 1451)	Top 2 (**493, 765**)	Top 2 (**350, 164**)	5 (164, 169, 350, 493, 765)	89.7
Top 2 (169, **1451**)	Top 4 (493, 765, 377, **1423**)	Top 4 (**350, 164, 14, 1378**)	6 (14, 164, 350, 1378, 1423, 1451)	98.3
Top 4 (169, **1451, 1430, 1538**)	Top 4 (493, **765, 377, 1423**)	Top 4 (**350, 164, 14, 1378**)	10 (14, 164, 350, 377, 765, 1378, 1423, 1430, 1451, 1538)	96.6
Top 4 (169, **1451, 1430,** 1538)	Top 8 (**493, 765,** 377, 1423, 249, **245, 267, 66**)	Top 8 (**350, 164, 14,** 1378, **43,** 976, **1325,** 353)	12 (14, 43, 66, 164, 245, 267, 350, 493, 765, 1325, 1430, 1451)	98.3

The numbers in bold are the genes selected by the GA from the feature pool

Table 4.3 shows the experimental results of applying our algorithm to Colon cancer data. To show the robustness of GA approach, we test several feature pools, each of which contains a different number of top-ranked features chosen from the three methods. Features finally selected by the genetic algorithm are highlighted. In the table, the genetic algorithm selects a subset of 6 genes and achieves 98.3% highest accuracy while SVM-RFE needs 16 genes to get the same accuracy. In another case, our approach finds a subset of 12 genes from a different gene pool,

which also reaches 98.3% accuracy. From Table 4.3, we can see that our method is capable of selecting smaller sized feature subsets with the highest accuracy.

4.4 Experiment on Prostate Cancer Data

We further test our approach on Prostate cancer data set [89$, 39$]. This data set consists of training data and testing data. The training data contain 52 prostate tumor samples and 50 non-tumor (normal) prostate samples with 12600 genes. The testing data contain 34 samples (25 tumor and 9 normal samples) obtained from a different experiment. With a test data set available in this experiment, 5-fold cross-validation is used to obtain the training accuracy.

In this experiment, we only use T-statistics and SVM-RFE to rank the features and input to the feature pool for the following reasons. From the previous experiment, we can see that the entropy-based feature selection does not produce results as good as the other two methods. Another reason is to save computation time.

Table 4.4 Top-20 Features from T-statistics and SVM-RFE on Prostate Cancer Data.

Feature Selection Algorithms	Top-20 Features
T-statistics	6185, 10138, 3879, 7520, 4365, 9050, **205**, 5654, 3649, **12153**, 3794, 9172, 9850, 8136, 7768, 5462, 12148, 9034, 4833, 8965
SVM-RFE	10234, **12153**, 8594, 9728, 11730, **205**, 11091, 10484, 12495, 49, 12505, 10694, 1674, 7079, 2515, 11942, 8058, 8658, 8603, 7826

The numbers in bold are the common gene(s)/feature(s) selected by two methods.

Table 4.4 presents the Top-20 features ranked by the two methods, in which there are only two common genes. Next, SVM classifies the test data based on these selected top features.

Table 4.5 demonstrates the training and testing accuracies from the two algorithms. SVM-RFE performs better in terms of higher training accuracy. The highest testing accuracy achieved by SVM-RFE is 94.1%, which is lower that the highest accuracy (97.1%) obtained by the other.

Since this data set is relatively large with 12600 features, we run the GA with smaller population size and fewer generations to reduce time consumption:

- Population size: 10
- Number of generations: 5
- Probability of crossover: 1
- Probability of mutation: 0.001

We adopt 5-fold validation for training accuracy. The results of our algorithm on the Prostate cancer data are presented in Table 4.6.

Table 4.5 Training and Testing Accuracy of T-statistics and SVM-RFE on Prostate Cancer Data

Top Features	Training Accuracy (%)		Testing Accuracy (%)	
	T-statistics	SVM-RFE	T-statistics	SVM-RFE
2	76.5	84.3	97.1	73.5
4	78.4	86.3	97.1	70.6
8	86.3	96.1	88.2	73.5
16	83.3	100.0	88.2	85.3
32	89.2	100.0	88.2	94.1
64	90.2	100.0	76.5	91.2
128	91.2	99.0	91.2	91.2
256	93.1	95.1	82.4	91.2
512	93.1	95.1	82.4	91.2
1024	91.2	94.1	85.3	94.1
2048	91.2	93.1	88.2	94.1
4096	89.2	92.2	94.1	94.1
8192	90.2	91.2	97.1	94.1
12600	89.2	91.2	97.1	94.1

For all of the three feature pools in Table 4.6, our approach can obtain the best testing accuracy (94.1%) with much smaller feature subsets compared to SVM-RFE. However, the testing accuracy is not as good as the one achieved by Top-2 or Top-4 genes selected by T-statistics. In the first case, our approach selects a subset of 3 genes, which can achieve 93.1% training accuracy, while SVM-RFE and T-Statistics cannot even reach the same training accuracy with 4 genes. In the second case, a subset of 4 genes selected by our GA achieves 95.1% training accuracy, which is higher than SVM-RFE and T-statistics with the same size of gene subset. In the last case, we select a subset of 8 genes, which achieves the same training accuracy as SVM-RFE does, but with higher testing accuracy.

Table 4.6 GA Experiments on Prostate Cancer Data

Gene/Feature Pool		Feature Subsets Selected by GA	Training Accuracy (%)	Testing Accuracy. (%)
T-statistics	SVM-RFE			
Top 2 (**6185, 10138**)	Top 2 (**10234**, 12153)	3 (6185, 10138, 10234)	93.1	94.1
Top 4 (**6185**, 10138, **3879**, 7520)	Top 4 (**10234**, 12153, **8594**, 9728)	4 (3879, 6185, 8594, 10234)	95.1	94.1
Top 8 (6185, 10138 , 3879, 7520, 4365 9050, **205,** 5654)	Top 8 (**10234, 12153, 8594**, **9728**, **11730, 205, 11091**, **10484**)	8 (205, 8594, 9728, 10234, 10484, 11091, 11730, 12153)	96.1	94.1

The numbers in bold are the genes selected by the GA from the feature pool.

In this chapter, we carried out two experiments on microarray datasets using our HGFS framework, in which a simple genetic algorithm is used to optimize the classification accuracy without explicit control of the size of feature subsets. We will use a new genetic algorithm that is designed to achieve multiple goals in the next chapter.

Chapter 5

Using A Genetic Algorithm With Size Control

Different feature selection algorithms can have different objectives that they aim to optimize. For example, the algorithms can search a feature subset such that the accuracy of the induced classifier is maximal or sufficient. Alternatively, the algorithms can find a feature subset with the smallest dimensionality of which the classification accuracy exceeds a specified value. In this chapter, the objective of our HGFS framework is to obtain a balance between the size of feature subsets and classification accuracy. To achieve this objective, we design a new genetic algorithm to control the size and the classification accuracy at the mean time.

5.1 A Different Fitness Function

We have introduced the HGFS framework in Chapter 4. In the model, we used a simple genetic algorithm to combine different feature selection criteria. The genetic algorithm was designed to optimize only one objective: the classification accuracy of the selected feature subset. There is no size control when selecting feature subsets. If there are several feature subsets that achieve the same classification accuracy, the genetic algorithm can not guarantee to find the feature subset of smallest size. Furthermore, sometimes it is acceptable that sacrificing a certain degree of accuracy for a smaller size of feature subsets. Thus, we change the simple genetic algorithm by modifying the fitness function to obtain a balance between the size and the accuracy of the feature subset.

Fitness Function

The new genetic algorithm is designed to achieve two objectives: maximize classification accuracy of the feature subset and minimize the number of features selected. To do so, we define the following fitness function:

$$F(i) = w * C(i) + (1 - w) * (1 / S(i))$$

where i is a feature vector representing a feature subset selected and w is a parameter between 0 and 1. The function is composed of two parts. The first part is a weighted classification accuracy $C(i)$ from the classifier and the second part is weighted size $S(i)$ of the feature subset represented by i. For a given w, the fitness of an individual i is increased as the classification accuracy of i increases and decreased as the size of i increases. Increasing the value of w means that we give more priority on the accuracy over the size. On the other hand, reducing the value of w will give more penalty on the size of i. By adjusting w, we can achieve a tradeoff between the accuracy and the size of the feature subset obtained.

For genetic operators (selection, crossover and mutation), we used the same ones described in Chapter 3. The new genetic algorithm is applied on Colon Cancer and Prostate Cancer data with the same parameters used in Chapter 4.

5.2 Experiment on Colon Cancer Data

Table 5.1 shows the feature subsets selected by the GA and the classification accuracy of the subsets on the data. We test several feature pools (no more than 20 features in total) with different values of parameter w in the fitness function. Each feature pool contains a different number of top-ranked genes from the three methods (Entropy-based, T-statistics, and SVM-

RFE). The results demonstrate that the feature subsets selected by our GA can accomplish the two goals: achieve either higher accuracy with smaller size or equal accuracy with smaller size compared to the feature subsets of the same size level selected by the other three.

As we can see from the table, reducing *w* does affect the size of feature subsets selected. Smaller values of *w* impose more penalties on the size of the subsets selected. Therefore using smaller *w* tends to select smaller subsets. In general, reducing *w* reduces the accuracy as well. However, there are a few exceptions in the table. For example, in the (4, 8, 8) feature pool, the GA chooses a subset of 9 features reaching 100% accuracy with $w = 0.75$. This subset is smaller than the one of 12 features with $w = 0.85$, but their accuracies are the same (100%). In addition, the subset obtains higher accuracy than the one of 10 features with $w = 0.8$. These indicate that there may exist redundancy, interaction and correlations between these features so that the feature subset with smaller size can achieve higher accuracy.

Table 5.1 LOO Accuracy of the New GA on Colon Cancer Data

w	Feature Pool*	GA	LOO Accuracy (%)
0.85	2, 4, 4	6 (14, 15, 70, 175, 249, 493)	96.6
	4, 4, 4	7 (14, 15, 70, 175, 249, 377, 493)	96.6
	4, 8, 8	12 (14, 15, 70, 164, 175, 245, 267, 377, 493, 1378, 1423, 1451)	100
0.8	2, 4, 4	3 (70, 175, 493)	91.9
	4, 4, 4	4 (14, 70, 493, 1430)	93.5
	4, 8, 8	10 (14, 15, 66, 70, 175, 245, 493, 1378, 1423, 1430)	98.4
0.75	2, 4, 4	2 (377, 1423)	88.7
	4, 4, 4	3 (14, 377, 493)	91.9
	4, 8, 8	9 (14, 15, 70, 175, 267, 493, 1430, 1451, 1538)	100
0.7	2, 4, 4	1 (377)	83.9
	4, 4, 4	2 (249, 377)	91.9
	4, 8, 8	3 (70, 267, 1451)	90.3

*The three numbers in a feature pool represent the number of top features selected from entropy-based, T-statistics and SVM-RFE respectively

Compared to Table 4.3, the result of using the simple genetic algorithm, the new genetic algorithm can find feature subsets with smaller size and higher accuracy. For example, in case of

w = 0.7, a subset with 2 features selected from feature pool (4, 4, 4) obtains 91.9 % accuracy, which is higher than the subset with 5 features does from Table 4.3 (89.7%). As another example, for the subset (9 features) with 100% accuracy in Table 5.1, it is smaller than the subset (12 features) with 98.3% accuracy in Table 4.3.

5.3 Experiment on Prostate Cancer Data

The three feature selection methods described in Section 3.3, that is, the entropy-based, T-statistic and SVM-RFE, are used to form the feature pool.

Table 5.2 Accuracies of Entropy-based, T-Statistic, SVM-RFE on Prostate Cancer Data

Top Features	Training Accuracy (%)			Testing Accuracy (%)		
	Entropy-based	T-statistics	SVM-RFE	Entropy-based	T-statistics	SVM-RFE
2	59.8	76.5	84.3	73.5	97.1	73.5
4	59.8	78.4	86.3	73.5	97.1	70.6
8	61.8	86.3	96.1	73.5	88.2	73.5
16	62.8	83.3	100	73.5	88.2	85.3
32	63.7	89.2	100	73.5	88.2	94.1
64	64.7	90.2	100	73.5	76.5	91.2
128	63.7	91.2	99.0	73.5	91.2	91.2
256	63.7	93.1	95.1	73.5	82.4	91.2
512	67.7	93.1	95.1	76.5	82.4	91.2
1024	68.6	91.2	94.1	73.5	85.3	94.1
2048	71.6	91.2	93.1	73.5	88.2	94.1
4096	76.5	89.2	92.2	82.4	94.1	94.1
8192	87.3	90.2	91.2	97.1	97.1	94.1
12600	89.2	89.2	91.2	97.1	97.1	94.1

Table 5.2 demonstrates the training accuracy and testing accuracy from the three algorithms. SVM-RFE performs better in terms of higher training accuracy. The highest testing accuracy achieved by SVM-RFE is 94.1%, which is lower than the highest accuracy (97.1%) obtained by the other two.

Again, we compare the top-20 features ranked from the three methods in Table 5.3 and find out that no genes are shared by the three. There are only two common genes (205 and 12153) shared by T-statistics and SVM-RFE.

Table 5.3 Top-20 Features From Entropy-based, T-Statistics, SVM-RFE on Prostate Cancer Data

Feature Selection Algorithms	Top-20 Features
Entropy-based	4234, 1058, 2789, 2474, 575, 4502, 6472, 12354, 5041, 3474, 727, 9994, 1585, 6365, 7249, 5823, 8052, 11401, 11926, 9926
T-Statistics	6185, 10138, 3879, 7520, 4365, 9050, **205**, 5654, 3649, **12153**, 3794, 9172, 9850, 8136, 7768, 5462, 12148, 9034, 4833, 8965
SVM-RFE	10234, **12153**, 8594, 9728, 11730, **205,** 11091, 10484, 12495, 49, 12505, 10694, 1674, 7079, 2515, 11942, 8058, 8658, 8603, 7826

Table 5.4 Training and Testing Accuracy of the New GA on Prostate Cancer Data.

w	Feature Pool*	GA	Training Accuracy (%)	Testing Accuracy (%)
	2, 4, 4	5 (4234, 6185, 7520, 8594, 10138)	93.1	94.1
0.85	4, 4, 4	7 (2474, 2789, 4234, 6185, 7520, 8594, 10234)	98.0	94.1
	4, 8, 8	10 (205, 3879, 4234, 5654, 6185, 7520, 10138, 10234, 11091, 11730)	99.0	94.1
	2, 4, 4	4 (3879, 6185, 9728, 10234)	92.2	94.1
	4, 4, 4	6 (2474, 2789, 4234, 6185, 7520, 10234)		
0.8			98.0	94.1
	4, 8, 8	8 (205, 8594, 9728, 10234, 10484, 11091, 11730, 12153)	96.1	94.1
	2, 4, 4	3 (6185, 10234, 12153)	89.2	94.1
0.75	4, 4, 4	3 (3879, 10234, 12153)	91.2	94.1
	4, 8, 8	4 (205, 3879, 9728, 10234)	91.2	94.1
	2, 4, 4	1 (6185)	85.3	94.1
0.7	4, 4, 4	2 (3879, 10234)	89.2	94.1
	4, 8, 8	3 (205, 8594, 10138)	91.2	94.1

*The three numbers in a feature pool represent the number of top features selected from Entropy-based, T-statistics and SVM-RFE respectively

Table 5.4 presents the results of applying the new GA on the Prostate Cancer data. From all the cases in the table, the GA obtains 94.1% testing accuracy, which is the highest one that can be reached by SVM-RFE. This testing accuracy is lower than the one obtained by the two feature subsets (with top-2 and top-4 features) selected by T-statistics. However, the training accuracies of these two feature subsets from T-statistics are very low. As to the entropy-based method, although it can also achieve 97.1% testing accuracy, it requires too many features. By reducing the value of parameter *w* associated with a feature pool, we can obtain a feature subset with smaller size. From the table, we can see that for a given feature pool, the accuracy is reduced as well in most cases when a smaller *w* is used. All the feature subsets selected by the GA from the feature pools achieve higher training accuracy than those subsets with equal or the next larger size from all the three methods.

5.4 Summary

We proposed a hybrid genetic feature selection (HGFS) framework in Chapter 3. Its components include a feature pool, a genetic algorithm and an induction algorithm. Feature pool are formed by collecting outputs from existing feature selection methods or even human experts. Then the genetic algorithm utilizes the induction algorithm that we chose to use SVM with linear kernel to evaluate every candidate feature subset generated from feature pool. Two genetic algorithms with different fitness functions are used for the framework. In Chapter 4, we coupled the framework with a simple genetic algorithm whose fitness function is to maximize the classification accuracy. This fitness function embodies the objective of obtaining the best classification performance. It does not explicitly control the size of feature subsets. The experiments are conducted on two gene selection problems (i.e. Colon Cancer and Prostate

Cancer). The performance of our framework is compared with T-statistics, SVM-RFE and/or the entropy-based feature selection methods, which are employed to build the feature pool as well. In Chapter 5, the HGFS framework is coupled with the second genetic algorithm with a different fitness function. This fitness function gives the control of both classification accuracy and the size of feature subsets to achieve a balance between them by adjusting a parameter *w*. We tested the framework for the same datasets and compared with the same feature selection methods as in Chapter 4. In both chapters, the experimental results demonstrate that our framework is capable of selecting feature subsets with higher classification accuracy and/or smaller size compared to each individual feature selection methods, that is, SVM-RFE, T-statistics and the entropy-based method.

Chapter 6

Text Categorization

The rapid growth of computer and Internet technologies makes huge quantities of text-based data (e.g., e-mails, online news and articles, newsgroups, Web pages, science papers) more commonly available. Given the enormous amounts of data, manually organizing these documents is too expensive and infeasible. Furthermore, manually built catalogues are costly to maintain. To handle and organize mass amounts of documents in an easier way, one of the dominant approaches nowadays is automated text categorization (TC—a.k.a. text classification) [87$, 108$, 22$, 36$, 101$, 107$].

As a combination of information retrieval (IR) technology and machine learning technology, TC has gained a booming interest from researchers and developers in both areas [79$, 60$, 3$, 50$, 7$, 85$]. It utilizes state-of-art machine learning techniques that automatically build a classifier by learning characteristics of different categories of documents through an inductive process. However, the common problem of high dimensionality of TC tasks makes most machine learning based TC algorithms infeasible [108$, 22$]. Applying dimensionality reduction techniques (i.e. feature selection or feature extraction) is beneficial for increasing scalability, reliability, efficiency and accuracy of text classification algorithms [49$]. In the next chapters, we are interested in applying feature selection techniques to improve learning algorithms in TC applications.

6.1 Introduction

Text categorization (TC) is the study of assigning predefined category labels to natural language documents based on their contents. The categories do not necessarily need to be exclusive. In other words, a document may or may not belong to more than one category. In fact, document may not belong to any categories at all. To train a learning algorithm, a set of training texts is annotated with correct labels by human experts. The training set is fed into the learning algorithms along with its labels. The learning algorithms generalize the training set by adjusting a number of internal parameters.

Text categorization is different from text clustering which is similar to unsupervised machine learning in terms of the absence of prior knowledge about category labels. Text clustering automatically identifies categories and groups documents into clusters that have similar contents based on certain similarity measurements.

There are many potential applications of text categorization. The following are a few examples of its applications.

Document Organization

An instance of document organization is document indexing with a controlled dictionary, such as the ACM Classification Scheme [88$]. This is an automatic indexing of scientific articles by means of a controlled dictionary, where the categories are the entries of the controlled dictionary. In the case of digital libraries, documents are usually indexed by thematic metadata that describe their semantics with controlled vocabulary (e.g. keywords, key phrases, bibliography codes). Another example of document organization is classification of News articles and ads. This type of problem can be tackled by TC techniques.

Spam filtering

A persistent problem for Internet service providers and users is the deluge of spam, the unsolicited bulk messages indiscriminately sent by spammers. Because of the huge volume of junk mail, extra capacity or cost has to be added to handle the flood. The most widely recognized form of spam is e-mail spam. Text classification systems [19$, 83$, 77$, 97$] can classify incoming e-mail as negative (non-spam) or positive (i.e. spam) and reject those that they finds to be spam. A challenge with spam filtering applications is the lack of negative examples. While spam messages are everywhere, non-spam messages are hard to collect because of the privacy issues. The unbalanced distribution of data examples should be addressed by TC algorithms.

Hierarchical categorization of Web pages

Due to the tremendous increase of the amount of Web pages or sites, it is more and more difficult to find the information we are interested in. Classifying Web pages or sites under hierarchical catalogues can make a Web search easier by restricting the search to a particular category of interest. While manual categorization of Web pages is infeasible and costly to maintain, TC methods [10$, 104$] can be employed to do the job automatically.

6.2 Single-label and Multi-label Text Categorization

Depending on applications, there are two types of text categorization: single-label and multi-label text categorization. For instance, spam filters only concerns spam and non-spam and news filters only deal with two classes, namely relevant and irrelevant. However, other sources of textual data, such as news articles, e-mail, and digital libraries, are composed of multiple topics. It is often the case that documents are relevant to more than one topic.

Let $D = \{d_1, d_2, \ldots, d_N\}$ be a collection of documents and $W = \{ w_1, w_2, \ldots, w_M\}$ be the vocabulary, that is, a set of distinct terms contained in D. Let $C = \{ c_1, c_2, \ldots, c_{|C|}\}$ be a set of predefined categories or classes. The notation d_j represents a document from D. Similarly, w_k is a term from W. The parameters N and M are the total number of documents and words respectively. A TC system assigns a value (T for true and F for false) to each pair of $<d_j, c_i>$ to indicate if a document d_j belongs to category or class c_i.

In a multi-label TC task, each document can be assigned with any number of categories from a set of predefined categories, while a single-label TC task assigns exactly one category to each document. A special case of single-label TC is called binary TC, which classifies documents into two disjoint categories c and its complement $\bar{c}$. For example, a spam filtering system is trained to classifying each incoming email into spam (c) or non-spam ($\bar{c}$). Under the assumption that categories are stochastically independent of each other, a multi-label TC can be transformed into $|C|$ independent (disjoint) binary TC problems for each c_i and $\bar{c}_i$ ($i = 1, \ldots |C|$). For this reason, most of TC researchers focus on binary classification [87$]. Binary classifiers are built for each class in these systems. To classify a new document, one needs to apply all the binary classifiers and combine their decisions into a single decision. We limit to binary TC as well in the dissertation.

6.3 Text Categorization Process

A typical text categorization consists of five steps as shown in Figure 6.1: linguistic preprocessing, text representation, dimensionality reduction, classifier learning and classifier evaluation.

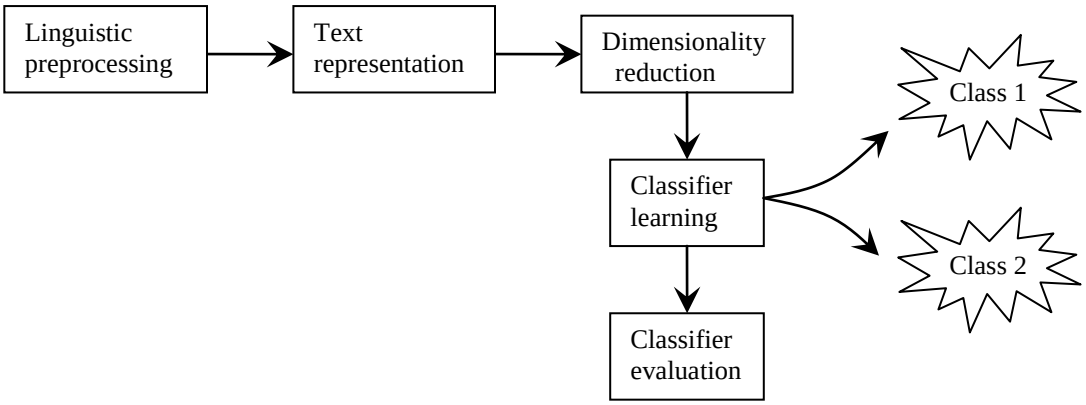


Figure 6.1 A typical process of text categorization

6.3.1 Linguistic Preprocessing

Linguistic preprocessing usually involves the removal of stopwords and word stemming. The stopwords are frequent words that do not carry any information about document topic, such as articles, prepositions, conjunctions, pronouns etc. Word stemming reduces words to their stems by using some suffix stripping procedure. The Porter stemmer [73$] is a well-known stemming algorithm. [23$] presents several stemming algorithms as well. Linguistic preprocessing is often performed to reduce the dimensionality of the feature space and the stochastic dependence between features. However, stemming and stopwords removal have been reported to hurt effectiveness sometimes [60$, 3$].

6.3.2 Text Representation

Text representation (also called document indexing) maps a text into a compact representation of its contents that can be directly interpreted by a classifier [88$]. Bag-of-words

model [85$] is the most popular one for document representation in TC field. This model represents a document as a sequence of terms (i.e. words) with the assumption that the words are independent of each other. Given a collection of training documents, a document d_j is represented as a vector of term weights $d_j = \langle w_{1j}, w_{2j}, \ldots, w_{Mj} \rangle$, where M is the number of words that appear in the collection. Each term is taken as a feature. Each term weight measures how much the term contributes to the classification of the document. It has been found that representations more sophisticated (e.g. using phrases rather than individual words as indexing terms) do not yield significantly better effectiveness [50$]. Therefore, we will adopt the bag-of-words model to represent documents in the experiments.

The document representation can use different sets of words [12$]. One consists of words that belong to each category isolated from the rest, which is known as local lexicum. On the other hand, global lexicum uses words from all categories to represent documents. In our work, global lexicum method is used for simplicity although it is claimed that local lexicum can produce better performance [65$].

There are different ways of determining term weights in a document. Boolean weighting or binary weighting assigns weight to 1 if the term w_k is present in the document d_j and 0 otherwise. Another simple weighting approach is called term frequency that uses the number of times w_k occurs in d_j as its weight. These two weight schemes do not consider frequency or distribution of the term throughout all documents in the collection. The *tfidf* (term frequency and inverse document frequency) weighting [84$] is a well-known approach for computing term weights (see Eq. 16). It assigns the weight to term w_k in document d_j in proportion to the term frequency in d_j, and in inverse proportion to document frequency of w_k which is the number of documents in the collection where w_k occurs at least once.

$$tfidf(w_k, d_j) = tf(w_k, d_j) \cdot \log(\frac{N}{n_k}) \quad (16)$$

where $tf(w_k, d_j)$ denotes the term frequency of w_k, and n_k denotes the document frequency of w_k. The function embodies the intuitions: (a) the more often a term occurs in a document, the more it is representative of contents of the document, and (b) the more documents a term occurs in, the less discriminating it is [87$]. The *tfidf* weights are usually normalized to consider documents of different lengths by cosine normalization:

$$w_{kj} = \frac{tfidf(w_k, d_j)}{\sqrt{\sum_{s=1}^{M} (tfidf(w_s, d_j))^2}} \quad (17)$$

There exist many variants of *tfidf* weighting that differ from each other in terms of logarithms, normalization or other correction factors [84$].

A slightly different approach [7$] called *ltc*-weighting replaces the raw term frequency with the logarithm of the term frequency, thus reducing the effects of large differences in frequencies.

$$w_{kj} = \frac{\log(tf(w_k, d_j) + 1) \cdot \log(\frac{N}{n_k})}{\sqrt{\sum_{s=1}^{M} (\log(tf(w_s, d_j) + 1) \cdot \log(\frac{N}{n_k}))^2}} \quad (18)$$

This is the term weighting approach employed in our work.

6.3.3 Dimensionality Reduction

A major difficulty in TC problems is the high dimensionality of the feature space. Even a text collection with moderate size often has tens of thousands of features, which will be cost-prohibitive for many learning algorithms that do not scale well to large problem sizes. In

addition, it is known that most words are irrelevant for the classification task and some of them even introduce noise that may decrease the overall performance [85$]. Furthermore, due to the lack of sufficient training documents compared to the number of features (namely data sparsity) the results of the algorithms become unreliable. Thus, it is necessary to reduce dimensionality by applying feature selection (a.k.a. term selection in TC) or feature extraction (a.k.a. term extraction in TC) techniques. Feature selection reduce the dimensionality of the feature space by only retaining those most informative or discriminative terms, thus increasing computational efficiency and effectiveness and avoiding overfitting.

The wrapper approaches we mentioned in the previous chapters can be used for this purpose of feature selection. It repeatedly calls induction algorithms to evaluate all possible feature subsets. In general, wrapper methods have been shown to perform better than filters [22$, 29$]. However, due to the large size of term spaces and data examples, wrapper methods can be rather time-consuming and thus impractical for TC applications. For this reason, the use of faster and simpler filter approaches is prominent in TC area. These approaches measure the importance of a term according to a particular feature scoring metric and the best *k* terms are kept.

Local and global feature selection

Depending on the scope where feature selection is performed, there are two distinct ways of feature selection in TC domains: local feature selection and global feature selection. While the former chooses different term subsets for the classification of each category, the latter selects a fixed set of terms for the classification under all categories. Most proposed techniques can be used for both local and global feature selection.

Normally feature selection functions are defined “locally” for one category. Terms are assessed and obtained a category-specific score for each category. Let $s(w_k, c_i)$ be the score of

term w_k for class c_i. In order to assess the value of a term for all categories in a global sense, there are several ways to do the globalization. For instances, we can either use the sum $s_{sum} = \sum_{i=1}^{|C|} s(w_k, c_i)$ [108$, 1$], or the weighted sum $s_{sum} = \sum_{i=1}^{|C|} P(c_i) s(w_k, c_i)$ [108$, 25$], or the maximum $s_{sum} = \max_{i=1}^{|C|} s(w_k, c_i)$ [108$, 80$] to compute the global scores of terms. In this dissertation, we choose to do global feature selection in the experiments.

6.3.4 Classifier Learning

Many information retrieval, statistical classification and machine learning techniques have been applied to TC domains. Examples are Rocchio's algorithm [87$], regression models [108$], K-nearest neighbor (KNN) [108$], naïve Bayes [101$], SVM [22$, 36$, 101$], Decision trees (e.g. C4.5 decision tree algorithm [36$]), and neural networks [107$] etc.

Rocchio's algorithm

Rocchio is the classic method for document routing or filtering in information retrieval. Rocchio classifiers assume that each prototype vector representing a particular category must combine the properties of both positive and negative example documents. Each representative vector (i.e. classifier) $\vec{c}_i = (w_{1i}, ..., w_{Mi})$ for category c_i are built by means of the following formula

$$w_{ki} = \beta \cdot \sum_{d_j \in c_i} \frac{w_{kj}}{|c_i|} - \gamma \sum_{d_j \in \bar{c}_i} \frac{w_{kj}}{|\bar{c}_i|}$$

where w_{kj} is the weight of term t_k in document d_j and β and γ are constants to control the relative importance of positive and negative examples. Classification is achieved by comparing the input feature vector against each of classifiers in turn according to some similarity measure such as a

cosine measure or Euclidean distance. The classifier is quite simple and fast. Rocchio method, as all linear classifiers, has the disadvantage that it divides the space of document linearly [87$].

K-nearest Neighbor

K-nearest neighbor assumes that the class labels of the most similar (nearest in feature space) neighbors can predict the class of a test document. The algorithm identifies the *k* closest points to the test point according to some similarity measure such as Euclidean distance and classifies the test document the same as the majority of the *k* nearest neighbor points. In case of a tie, the test document is assigned to the class of a closest point.

Decision Trees

Decision Trees are algorithms that generalize training data in the forms of a decision tree. Nodes in the tree represent features and branches are associated values of the corresponding features. The leaves of the tree correspond to classes. Explicit logical rules can be induced from the decision trees, which can be interpreted by user. A new instance is classified by checking features at the nodes of the tree and follows the branches corresponding to their observed values in the instance. Upon reaching a leaf, the class at the leaf is assigned to the instance.

SVMs have been introduced in Section 3.3 and the naïve Bayes classifier in our experiments will be explained later.

6.3.5 Classifier Evaluation

One of the traditional measures to evaluate classification effectiveness is accuracy, which describes the percentage of correct classification decisions. However, accuracy is not widely used in TC areas for the reason that the two categories c_i and $\bar{c}_i$ are usually unbalanced [88$].

Given a test set, accuracy for each category c_i can be computed by a contingency table (see Table 6.1) as:

$$A_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}$$

where TP_i (true positive for class c_i) is the number of test documents correctly classified under c_i by the classifier and FP_i (false positive for class c_i) is the number of test documents incorrectly classified under c_i by the classifier. FN_i (false negative for class c_i), and TN_i (true negative for class c_i) are defined accordingly. TC applications usually have large value of the denominator, which makes accuracy insensitive to variations in the number of correct decisions (i.e. $TP_i + TN_i$). Besides, if A is the adopted evaluation measure, in the frequent case of a very low average generality, the trivial rejector which labels every document d_j as false for each c_i tends to outperform all nontrivial classifiers [11$].

Table 6.1 The Contingency Table for Category c_i

Category c_i		The Truth	
		YES	**NO**
Classifier Decisions	**YES**	TP_i	FP_i
	NO	FN_i	TN_i

Instead, the evaluation of classification in TC applications is usually analyzed from multiple perspectives – precision, recall, and F-measure. Precision π measures the percentage of documents predicted to be in class c_i that in fact belong to it. Recall r is the percentage of documents truly belonging to c_i that are classified into this class. These two measures can be computed from Table 6.1 as well. According to their definitions, the precision and recall with respect to c_i can be calculated as:

$$\pi_i = \frac{TP_i}{TP_i + FP_i} \tag{19}$$

$$r_i = \frac{TP_i}{TP_i + FN_i} \tag{20}$$

For evaluating average performance across categories, there are two ways to obtain the overall precision (π) and recall (r) for all classes, namely micro-averaging and macro-averaging. Macro-averaging precision (or recall) is obtained by first evaluating locally for each category, and then globally averaging over the results of different categories:

$$\pi^A = \frac{\sum_{i=1}^{|C|} \pi_i}{|C|} \tag{21}$$

$$r^A = \frac{\sum_{i=1}^{|C|} r_i}{|C|} \tag{22}$$

On the other hand, micro-averaging is acquired by summing over all individual contingency tables:

$$\pi^I = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FP_i)} \tag{23}$$

$$r^I = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FN_i)} \tag{24}$$

There is an important distinction between the two types of averaging. Macro-averaging gives equal weight to each category, while micro-averaging gives equal weight to every document. Therefore, the performance on categories with low generality (i.e. few training examples) will be emphasized by macro-averaging. Which averaging method should be chosen depends on the application.

Usually there is a trade-off between recall and precision. That is, to achieve higher level of recall often means sacrificing precision and vice versa [87$]. Sometimes it is better to evaluate a classifier by means of a measure that combines recall and precision. F-measure is one of the measures proposed to combine them through a function [79$].

$$F_\beta = \frac{(\beta^2 + 1)\pi r}{\beta^2 \pi + r} \qquad (0 \le \beta \le \infty) \tag{25}$$

where β may be seen as the relative degree of importance attributed to π and r. When β is set to 1, equal importance is given to precision and recall. This is called F_1-measure, which is the most popular combination currently being used.

$$F_1 = \frac{2\pi r}{\pi + r} \tag{26}$$

Similarly, micro-averaged F_1 (F_1^I) and macro-averaged F_1 (F_1^A) is defined as:

$$F_1^A = \frac{\sum_{i=1}^{|C|} F_{1i}}{|C|} \tag{27}$$

$$F_1^I = \frac{2 \cdot \sum_{i=1}^{|C|} TP_i}{2 \cdot \sum_{i=1}^{|C|} TP_i + \sum_{i=1}^{|C|} FP_i + \sum_{i=1}^{|C|} FN_i} \tag{28}$$

Chapter 7

Feature Selection for Text Categorization

Since a huge number of terms are often involved in TC applications, feature selection is essential to make the learning task more efficient and accurate. We proposed a simple feature selection method called Relation Strength and Frequency Variance (RSFV) measure based on the hypothesis that informative features are those that are highly correlated with the class and distribute most differently among all classes. We carry out experiments on two standard text corpora and analyze the results from each of the perspectives of micro-averaged and macro-averaged precision, recall and F_1-measure, since each serves different purposes. The experiments reveal that RSFV is also effective in that it obtains comparable results and outperforms other traditional methods in many situations.

7.1 Related work

Many various feature scoring metrics have been proposed in TC areas. In [108$], Yang and Pedersen gave a thorough review of five feature selection methods: Document Frequency, Information Gain, Mutual Information, χ^2-statistics, and Term Strength. Mladenic and Grobelnik introduced odds ratio in [63$]. Sebastiani and Simi [25$] proposed a modification of χ^2, called simplified χ^2. A method called BNS (Bi-Normal Separation) is proposed by Forman [22$]. Several widely used feature selection methods in TC domains are briefly described in the next.

Document Frequency (DF) and Term Frequency (TF)

DF for a term is the number of documents in which the term occurs, while TF is the total number of appearances of a term. They are one of the simplest feature selection techniques. Based on the assumption that rare terms are non-informative for category prediction or not influential in global performance, DF (TF) retains only the terms whose document (term) frequency is no less than some predefined threshold. DF or TF is usually considered an ad hoc approach to improve efficiency, not a principled criterion for selecting predictive features [108$]. There are several variants of DF or TF often adopted by many researchers in the linguistic preprocessing stage of TC. One is to remove all terms occurring in at most *x* (with popular range from 1 to 3) training documents [20$]. Another is to remove all terms that occur at most *x* times (with popular range from 1 to 5) in the training set [3$, 35$].

Information Gain (IG)

Information gain scoring metrics is based on information theory. It measures the decrease in entropy by knowing the presence or absence of a feature (i.e. a term) in a document. The following equation defines the information gain of a term *w* over all categories:

$$IG(w) = -\sum_{i=1}^{|C|} P(c_i)\log P(c_i) + P(w)\sum_{i=1}^{|C|} P(c_i \mid w)\log P(c_i \mid w) + P(\overline{w})\sum_{i=1}^{|C|} P(c_i \mid \overline{w})\log P(c_i \mid \overline{w}) \quad (29)$$

where

- $P(c_i)$ is the probability that a random document belongs to c_i.
- $P(w)$ is the probability of the occurrence of the word *w* in a random document.
- $P(c_i \mid w)$ is the probability that a random document belongs to class c_i if *w* occurs in it.
- $P(\overline{w})$ is the probability of that *w* does not appear in a random document.

- $P(c_i \mid \overline{w})$ is the probability that a random document belongs to class *c* if *w* does not appear in it.

In our experiments, we estimate $P(c_i)$ as the percentage of documents in the total collection that belongs to class c_i. For *P*(w), it can be estimated as the percentage of documents in which the word *w* occurs and $P(\overline{w})$ is estimated accordingly. Moreover, $P(c_i \mid w)$ can be computed as the fraction of documents from class c_i that have at least one occurrence of word w and $P(c_i \mid \overline{w})$ is estimated in the same way.

A disadvantage of IG pointed out by [70$] is that it grows not only with the increase of dependence between *w* and *c*, but also with the increase of the entropy of *w*. That is why features with low entropy receive smaller IG weights, although they may be strongly correlated with a class.

Mutual Information (MI)

MI is an information-theoretic measure of association between the word and class. Pointwise MI between term *w* and a category *c* is defined to be:

$$MI(w,c) = \log \frac{P(w,c)}{P(w)P(c)} \tag{30}$$

where *P*(*w*, *c*) is the joint probability that a random document contains term *w* and belongs to category *c*. Moreover, MI has an equivalent form as:

$$MI(w,c) = \log P(w \mid c) - \log P(w) \tag{31}$$

As we can see from Eq. (31), one weakness of MI is that it tends to score rare terms (i.e. with smaller value of *P*(*w*)) higher than common terms given the terms with equal conditional probability *P*(*w* | *c*).

Odds Ratio (OR)

Odds ratio evaluates a term on the basis of its association with a set of positive documents, that is, the documents contain the term. Odds ratio is defined as:

$$OR(w,c) = \frac{P(w|c)\cdot(1-P(w|\bar{c}))}{(1-P(w|c))P(w|\bar{c})} \tag{32}$$

As we can see from the definition, OR does not consider those documents that do not contain *w*. Therefore, it ignores possible useful information about the correlation between the term and the class that is provided by those documents.

χ^2-statistics (CHI)

The χ^2-statistics measures the lack of independence between *w* and *c*. If *w* and *c* are independent, CHI has the lowest value of zero.

$$\chi^2(w,c) = N \cdot \frac{[P(w,c)P(\overline{w},\overline{c}) - P(w,\overline{c})P(\overline{w},c)]^2}{P(w)P(\overline{w})P(c)P(\overline{c})} \tag{33}$$

As a statistical test, it is known to behave erratically for very small expected counts, which are common in text classification both because of having rarely occurring word features, and sometimes because of having few positive training examples for a category [22$].

7.2 Relation Strength and Frequency Variance (RSFV) Measure

In this section, we will describe the feature selection metric (called RSFV) proposed by us. From the feature selection methods described above, we can see that most feature selection functions try to capture the correlations between terms and classes. They score terms based on one common principle: the more correlated terms are with category *c*, the more discriminative

terms are. To help find such correlations, it is good to introduce the two-way contingency table (see Table 7.1) for a term *w* and category *c*.

Table 7.1 The Contingency Table for Term *w* and Category *c*

	c	$\bar{c}$
w	A	B
$\bar{w}$	C	D

The contingency table records co-occurrence statistics for terms and categories. These statistics are also useful for estimating the probabilities in the definitions of previous feature selection functions.

- *A* denotes the number of documents of category *c* containing *w*.
- *B* denotes the number of documents that contain *w* but do not belong to *c*.
- *C* denotes the number of documents of category c in which w does not occur.
- *D* denotes the number of documents that neither contain *w* nor belong to *c*.

Moreover, we have the number of documents in the collection, $N = A + B + C + D$. We consider *A* and *D* represent positive relation of *w* with category *c*, while B and D represent negative relation. Apparently, the larger values of *A* (i.e. co-occurrence) and *D* (i.e. co-non-occurrence) are, the stronger positive relation *w* has with *c*. Similarly, the larger values of *B* and *C* indicate the stronger negative relation between *w* and *c*. If *w* has a strong positive relation with *c*, we call it a positive feature. We can predict the membership (non-membership) of a document of *c* by the presence (absence) of *w*. On the other hand, if *w* has a strong negative relation with *c*, namely a negative feature, the absence (presence) of *w* can predict the membership (or non-membership) of a document of *c*. Negatively correlated features are considered for the reason that negative features are numerous and quite valuable in practical experience, given the large class skew (i.e.

usually |c| << $|\bar{c}|$). The importance of negative features is empirically confirmed in the literature. A term with either strong positive relation or negative relation is a feature with good discriminative power. The bigger difference between *AD* and *BC*, the better the feature is. We define Eq. 34 to embody this idea to measure the strength of the relation (*RS*) between *w* and *c*:

$$RS = (AD - BC)^2 \tag{34}$$

A term with bigger value of Eq. 34 means it is more distinguishable for the category.

From another point of view, the more different the terms distributed in various categories, the more discriminative the terms are. We believe how frequently a term is likely to appear in the documents from a category can also indicate the importance of the term. The idea is that most important features appear frequently in related documents. However, the same features should not appear often in other categories. For example, thematic keywords usually occur frequently in the documents from the same category but rarely occur in other categories with different themes. Let E_{wc} be the expected term frequency of *w* in *c* and D_w to be the variance of the expected term frequency of *w* across all the categories.

$$E_{wc} = \frac{\sum_{d \in c} tf(w,d)}{|c|} \tag{35}$$

$$D_w = \frac{\sum_{c \in C} (E_{wc} - \mu)^2}{|C|} \tag{36}$$

where *tf*(*w*, *d*) is the term frequency of *w* in document *d* and μ is the expected value of E_{wc}. D_w demonstrates the degree of variability of the distributions of *w* among categories. The bigger value of D_w, the more distinguishable among categories the term *w* is.

Finally, we combine the above ideas and define the RSFV feature selection function in the following way:

$$RSFV(w,c) = \frac{D_w}{1+D_w} \cdot \log(RS+1) = \frac{D_w}{1+D_w} \cdot \log((AD-BC)^2+1) \quad (37)$$

where *A, D, B, C* and D_w are obtained from Table 7.1 and Eq. 36. The range of the scores obtained from the above formula is from 0 to ∞. We rank features in descending order based on the values of Eq. 37.

7.3 Experimental Setup

In order to test the performance of proposed RSFV feature selection method, we conduct experiments on two standard text categorization datasets. Two classic methods, information gain (IG) and odds ratio (OR), are tested on the same datasets for comparison with RSFV.

7.3.1 The Text Corpora

Two different datasets are used in the experiments: the Reuters-21578 corpus and the 20 Newsgroup corpus.

Reuters-21578 Corpus

The Reuters-21578 collection [48$] and its earlier variants has been a standard benchmark for TC tasks for many years. It is a set of 21,578 news stories published by Reuters in 1987, which are classified according to 135 thematic categories mostly concerning business and economy. Standard splits are defined by the creators of the collection to create various subsets of the corpus. Different splits have been used by researchers to test their systems. Majority of

researchers used ModApte split that selects 9,603 training documents and the other 3,299 test documents from 90 categories. We choose to use this split in our experiments too.

The distribution of documents across the categories is highly unbalanced, in the sense that some categories have few documents classified under them while others have thousands. For instance, the category “acq” contains 2369 documents in total with 1650 training documents and 719 test documents, while the category “castor-oil” only contains two documents in total with one for training and one for test. This characteristic makes it more challenging for machine learning techniques. The words in the corpus are little scattered, since almost half (49.91 percent) of the words appear in only one category and 16.25 percent in only two categories [12$]. The following document from category “coffee” gives an illustration of this text corpus.

```
ICO PRODUCERS TO PRESENT NEW COFFEE PROPOSAL

   LONDON, Feb 26 - International Coffee Organization, ICO,
producing countries will present a proposal for reintroducing
export quotas for 12 months from April 1 with a firm
undertaking to try to negotiate up to September 30 any future
quota distribution on a new basis, ICO delegates said.
   Distribution from April 1 would be on an unchanged basis as
in an earlier producer proposal, which includes shortfall
redistributions totalling 1.22 mln bags, they said.
   Resumption of an ICO contact group meeting with consumers,
scheduled for this evening, has been postponed until tomorrow,
delegates said.
```

20 Newsgroup

This collection (available at http://www.ai.mit.edu/people/jrennie/20Newsgroups/) contains 19997 (approximately 20000) articles nearly evenly partitioned among 20 different UseNet discussion groups. Except for a small fraction of the articles, each document belongs to exactly one newsgroup. Some of the newsgroups are very closely related to each other, while others are

highly unrelated. For example, three of them are about politics (named talk.politics.*) and two of them are about sports (named rec.sport.*). There is no independent test set available for this collection. Compared with the previous corpus, it includes a larger vocabulary and words typically have more meanings. Moreover, the e-mail style writing with UseNet header is very different from the Reuters corpus.

An example document from category “comp.graphics” is given in the following.

```
Xref: cantaloupe.srv.cs.cmu.edu comp.graphics:37261 alt.graphics:519
comp.graphics.animation:2614
Path: cantaloupe.srv.cs.cmu.edu!das-
news.harvard.edu!ogicse!uwm.edu!zaphod.mps.ohio-
state.edu!darwin.sura.net!dtix.dt.navy.mil!oasys!lipman
From: lipman@oasys.dt.navy.mil (Robert Lipman)
Newsgroups: comp.graphics,alt.graphics,comp.graphics.animation
Subject: CALL FOR PRESENTATIONS: Navy SciViz/VR Seminar
Message-ID: <32850@oasys.dt.navy.mil>
Date: 19 Mar 93 20:10:23 GMT
Article-I.D.: oasys.32850
Expires: 30 Apr 93 04:00:00 GMT
Reply-To: lipman@oasys.dt.navy.mil (Robert Lipman)
Followup-To: comp.graphics
Distribution: usa
Organization: Carderock Division, NSWC, Bethesda, MD
Lines: 65


                CALL FOR PRESENTATIONS

      NAVY SCIENTIFIC VISUALIZATION AND VIRTUAL REALITY SEMINAR

                Tuesday, June 22, 1993

         Carderock Division, Naval Surface Warfare Center
           (formerly the David Taylor Research Center)
                  Bethesda, Maryland

SPONSOR: NESS (Navy Engineering Software System) is sponsoring a
one-day Navy Scientific Visualization and Virtual Reality Seminar.
The purpose of the seminar is to present and exchange information for
Navy-related scientific visualization and virtual reality programs,
research, developments, and applications.

PRESENTATIONS: Presentations are solicited on all aspects of
Navy-related scientific visualization and virtual reality.  All
current work, works-in-progress, and proposed work by Navy
organizations will be considered.  Four types of presentations are
available.
```

```
     1. Regular presentation: 20-30 minutes in length
     2. Short presentation: 10 minutes in length
     3. Video presentation: a stand-alone videotape (author need not
   attend the seminar)
     4. Scientific visualization or virtual reality demonstration (BYOH)

Accepted presentations will not be published in any proceedings,
however, viewgraphs and other materials will be reproduced for
seminar attendees.

ABSTRACTS: Authors should submit a one page abstract and/or videotape to:

     Robert Lipman
     Naval Surface Warfare Center, Carderock Division
     Code 2042
     Bethesda, Maryland  20084-5000

     VOICE (301) 227-3618;  FAX (301) 227-5753
     E-MAIL  lipman@oasys.dt.navy.mil

Authors should include the type of presentation, their affiliations,
addresses, telephone and FAX numbers, and addresses.  Multi-author
papers should designate one point of contact.

DEADLINES: The abstact submission deadline is April 30, 1993.
Notification of acceptance will be sent by May 14, 1993.
Materials for reproduction must be received by June 1, 1993.

For further information, contact Robert Lipman at the above address.

     PLEASE DISTRIBUTE AS WIDELY AS POSSIBLE, THANKS.



Robert Lipman                       | Internet: lipman@oasys.dt.navy.mil
David Taylor Model Basin - CDNSWC   |       or: lip@ocean.dt.navy.mil
Computational Signatures and        | Voicenet: (301) 227-3618
   Structures Group, Code 2042      | Factsnet: (301) 227-5753
Bethesda, Maryland  20084-5000      | Phishnet: stockings@long.legs

The sixth sick shiek's sixth sheep's sick.
```

7.3.2 Classifier

Although known for their simplicity, naïve Bayes (NB) classifiers have been found to perform surprisingly well in information retrieval [101$, 60$, 35$]. NB classifiers assume that

all attributes of the training examples are independent of each other given the context of the class. While this naïve Bayes assumption is clearly violated in most real-world problems, NB classifiers often produce effective performance. Although other classifiers, such as SVM, are shown to be superior to NB classifiers [107$], they are more complicated and computationally intense than NB. When facing large number of attributes of TC tasks, NB classifiers are especially attractive to many researchers because of their efficiency.

In TC community, there are two types of NB classifiers in common use that are based on two different generative models [60$]. One model is called multi-bernoulli model. In this model, a document is represented by a vector of binary attributes indicating which words occur and do not occur in the document. The document is taken as the "event" and the absence or presence of words to be attributes of the event. The other model called multinomial event model represents a document by the set of word occurrences from the document. The individual word occurrences are considered to be the "events" and the document to be the collection of word events. When calculating the probability of a document, one multiplies the probabilities of the words that occur. The second model is what we used in our experiments.

According to Bayes' rule, to achieve the highest classification accuracy, a document d_j should be assigned to the class for which $P(c_i \mid d_j)$ is highest. That is:

$$C(d_j) = \arg\max_{c_i \in C} P(c_i \mid d_j) \tag{38}$$

Applying Bayes' theorem to Eq. (38), we can compute $P(c_i \mid d_j)$ from $P(c_i \mid d_j)$ (see Eq. 39):

$$P(c_i \mid d_j) = \frac{P(c_i)P(d_j \mid c_i)}{P(d_j)} \tag{39}$$

An assumption made is that a word's occurrence is independent of the other words in the document and the document length. So the estimation of $P(d_j \mid c_i)$ can be reduced to the estimation of $P(w_k \mid c_i)$:

$$P(d_j \mid c_i) = \prod_{k=1}^{|d_j|} P(w_k \mid c_i) \tag{41}$$

where $|d_j|$ is the number of words in d_j. To compute the value of $P(w_k \mid c_i)$, Laplace estimator suggested in [99$] is used:

$$P(w_k \mid c_i) = \frac{1 + tf(w_k, c_i)}{M + \sum_{s \in W} tf(w_s, c_i)} \tag{42}$$

where M is the total number of words and W is the whole set of features from the text collection. Combining the equations above, the final result of the NB classifier is the following:

$$\begin{aligned} C(d_j) &= \arg\max_{c_i \in C} \frac{P(c_i) \cdot \prod_{k=1}^{|d_j|} P(w_k \mid c_i)}{P(d_j)} \\ &= \arg\max_{c_i \in C} \frac{P(c_i) \cdot \prod_{w \in W} P(w \mid c_i)^{tf(w, d_j)}}{P(d_j)} \end{aligned} \tag{43}$$

The denominator can be discarded when computing the result, since it does not change the result. We assume that the category of a document does not depend on its length. $P(c_i)$ is estimated as the percentage of documents in the total collection that belongs to class c_i.

Because the naïve Bayes classifier assign d_j to the class for which $P(c_i \mid d_j)$ is highest, it means that each document is assigned to exact one category. This causes an interesting phenomenon that micro-averaged recall, precision and F_1 are all equal, which can be proved by the definition of micro-averaged recall, precision and F_1 (see Eq. 23, 24 and 28).

7.3.3 Feature Subset Size

There are several strategies to determine the size of feature subsets. One is to select a feature subset with a predefined number of features, which has been widely used [$57, $55, 9$]. Another common practice is to select features whose score are over a predefined threshold [22$]. The third strategy is to select a number of features proportional to the total number of features [64$, 108$]. The last two strategies are employed in our experiments.

7.4 Results

In this section, we present the results of our experiments for the two corpora under study. Training data are transformed into a word-document matrix in which each value is a word weight determined by *ltc*-weighting (see Eq. 18). Global feature selections are performed on both datasets. The globalization of feature scores is done by summing up local scores for each category. For the performance measurements, namely macro-averaged and micro-averaged recall, precision and F_1 (see Eq. 21 – 28), we use macro- (micro-) recall, precision, and F_1 as their short names respectively. Our system is implemented based on a toolkit named bow [61$].

7.4.1 Results on 20 Newsgroup

When we process this dataset with email style writing, UseNet headers, which include the subject lines, are skipped. Tokens are formed by contiguous alphabetic characters and stemming is performed at the same time. Stopwords are removed.

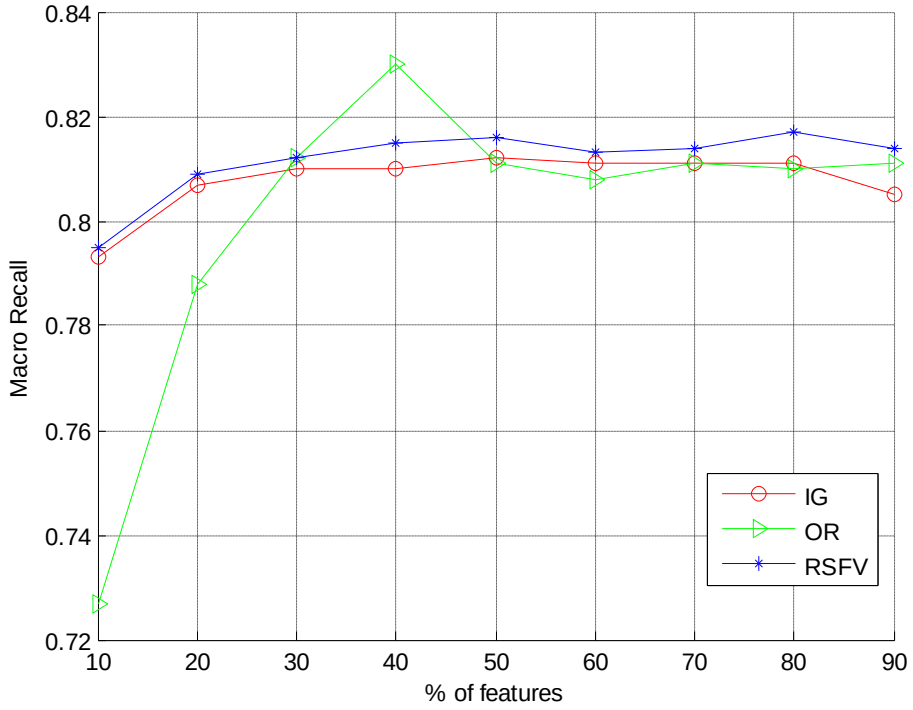


Figure 7.1 Macro-recall for 20 Newsgroup corpus

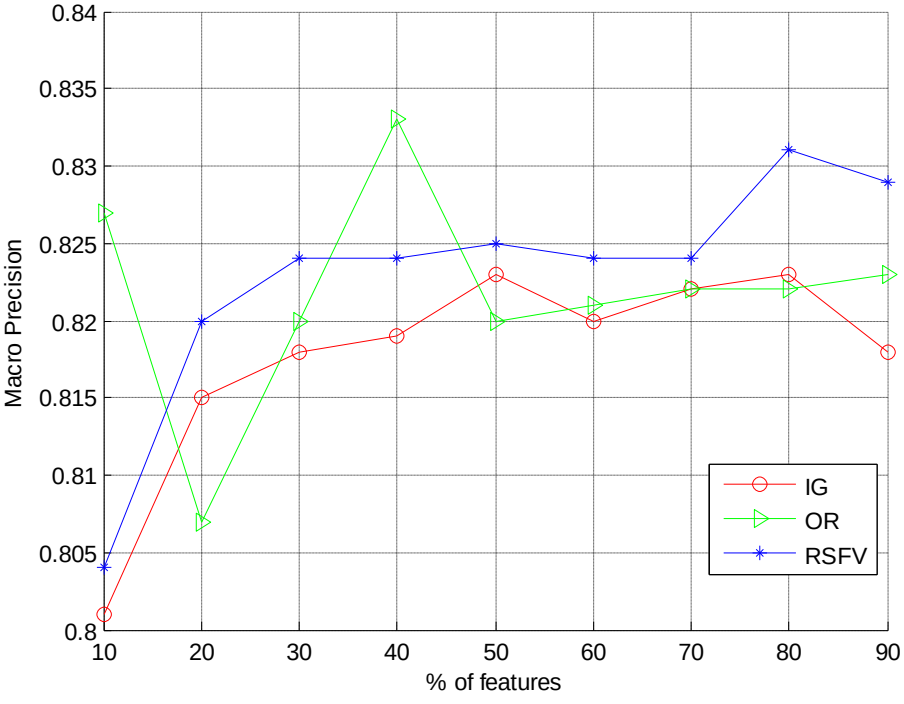


Figure 7.2 Macro-precision for 20 Newsgroup corpus

Since there is no separated test sets available, we randomly select a portion of the whole dataset as a test set. Five trials are carried out and the average performances (i.e. micro- and macro- recall, precision and F_1) of the trials are reported. In each trial, 20% of the data is held out for testing and the remaining is for training.

Figure 7.1 shows the average macro-recalls of the five trials. OR obtains the best macro-recall among the three with a feature subset of top 40% of features. This is 1.5% better than the macro-recall given by RSFV. However, it is worse than RSFV at the rest points. Especially at the point of 10% of features, RSFV achieves 6.8% higher value than OR. Compared to IG, RSFV is better to a small extent ($\leq 1\%$) at every point.

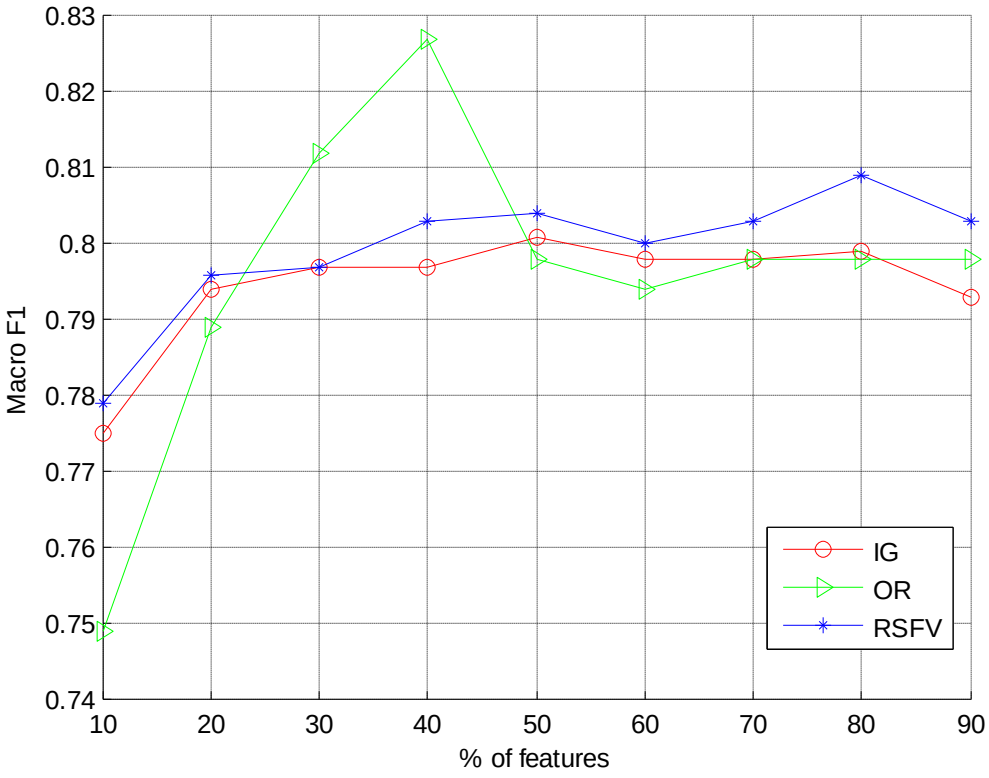


Figure 7.3 Macro-F_1 for 20 Newsgroup corpus

As shown in Figure 7.2, RSFV achieves slightly better macro-precision than IG with the increase no more than 1%. However, compared to macro-recall (in Figure 7.1), the increase of RSFV's macro-precision is more perceptible at every point. Moreover, RSFV produces higher macro-precision (up to 1.3% higher) than OR except at the two points (i.e. 10% and 40% of features), where OR gives 2.3% and 0.9% increase respectively.

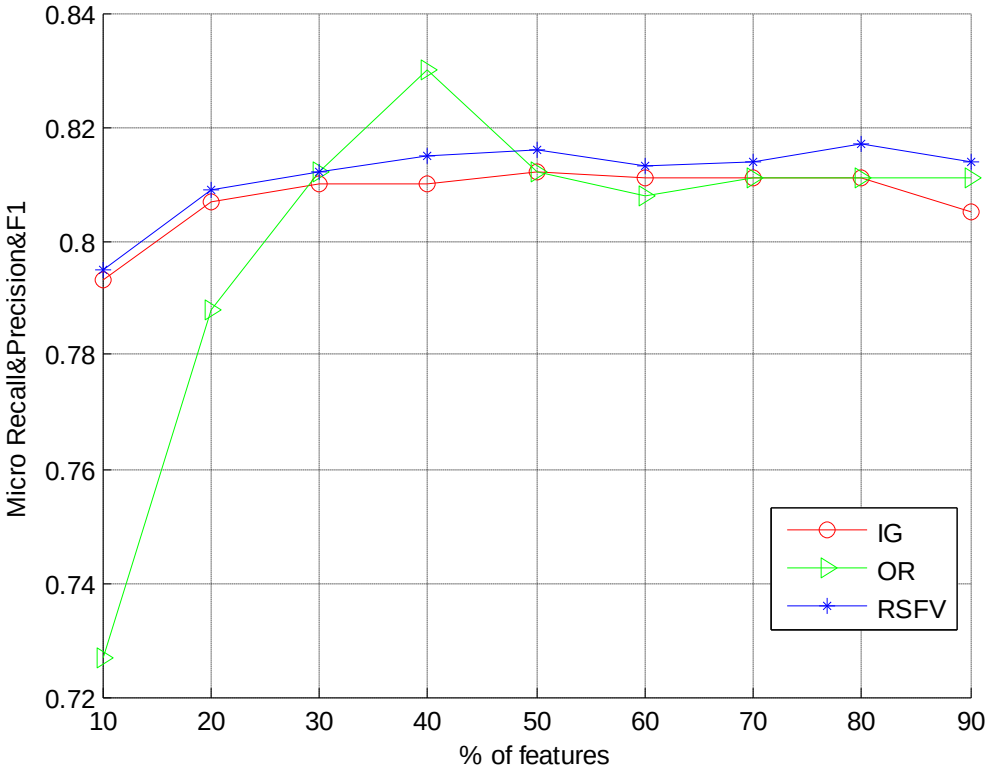


Figure 7.4 Micro-recall, precision, F_1 for 20 Newsgroup corpus

Figure 7.3 shows the comparison of the three methods by means of macro-F_1. Again, RSFV's performance is slightly better than IG. OR is 1.5% and 2.4% better than RSFV with the selection of 30% and 40% of features. However, for the rest selections of features, RSFV outperforms OR by up to 3% increase.

The micro-averaged recall, precision and F_1 of the three methods are shown in Figure 7.4. We can conclude that although OR can achieve best performances at one or two points, RSFV is the best in overall performances.

From the figures above, we can observe that the main trend of IG and RSFV is that the performances become better as the number of features increases. Their performances peak with 80% and 50% of features, respectively. As for OR, it peaks the performances with 40% of feature.

For a clearer view of the results of this experiment, Table 7.2 provides the exact results of applying IG, OR and RSFV on 20 Newsgroup text corpus, that is, Maro-averaged and Micro-averaged recall, precision, and F_1.

Table 7.2 The Results of IG, OR and RSFV on 20 Newsgroup

Features	IG				OR				RSFV			
	*MA. Re.	*MA. Pre.	*MA. F_1	*MIs	MA. Re.	MA. Pre	MA. F1	MIs	MA. Re.	MA. Pre.	MA. F_1	MIs
10%	0.793	0.801	0.775	0.793	0.727	0.827	0.749	0.727	0.795	0.804	0.779	0.795
20%	0.807	0.815	0.794	0.807	0.788	0.807	0.789	0.788	0.809	0.820	0.796	0.809
30%	0.810	0.818	0.797	0.810	0.812	0.820	0.812	0.812	0.812	0.824	0.797	0.812
40%	0.810	0.819	0.797	0.810	0.830	0.833	0.827	0.830	0.815	0.824	0.803	0.815
50%	0.812	0.823	0.801	0.812	0.811	0.820	0.798	0.812	0.816	0.825	0.804	0.816
60%	0.811	0.820	0.798	0.811	0.808	0.821	0.794	0.808	0.813	0.824	0.800	0.813
70%	0.811	0.822	0.798	0.811	0.811	0.822	0.798	0.811	0.814	0.824	0.803	0.814
80%	0.811	0.823	0.799	0.811	0.810	0.822	0.798	0.811	0.817	0.831	0.809	0.817
90%	0.805	0.818	0.793	0.805	0.811	0.823	0.798	0.811	0.814	0.829	0.803	0.814

*MA. Re: Macro-Recall *MA. Pre: Macro-Precision

*MA. F_1: Macro-F_1

*MIs: Micro-Recall, Micro-precision, and Micro-F_1

7.4.2 Results on Reuters-21578

In the preprocessing stage, tokens (words) are formed from contiguous alphabetic characters with no stemming. However, stopwords are removed. We first test the three methods with size of feature subsets (from 10 to 90 percent) proportional to the total number of features.

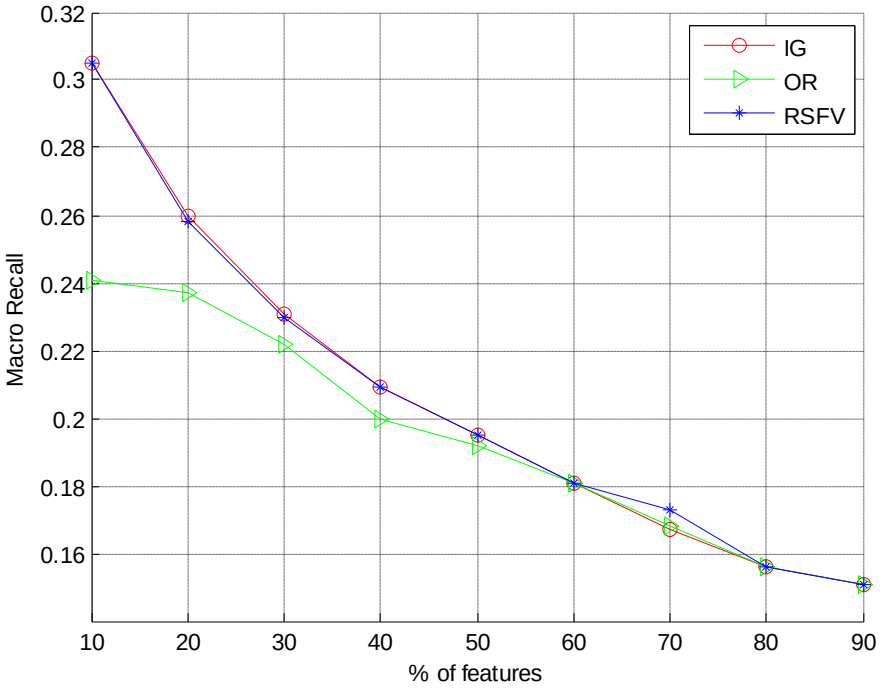


Figure 7.5 Macro-recall for Reuters with 10% – 90% features

Figure 7.5 shows the performance of RSFV is better than odds ratio (OR) with up to 6.4% increase. There is small difference (< 1%) between IG and RSFV with any selections of features. These two can be considered as equal in overall performance. For all of the three methods, their recall decreases as feature size increases. The best individual results are produced with only top 10% of features selected (i.e. 2135 features).

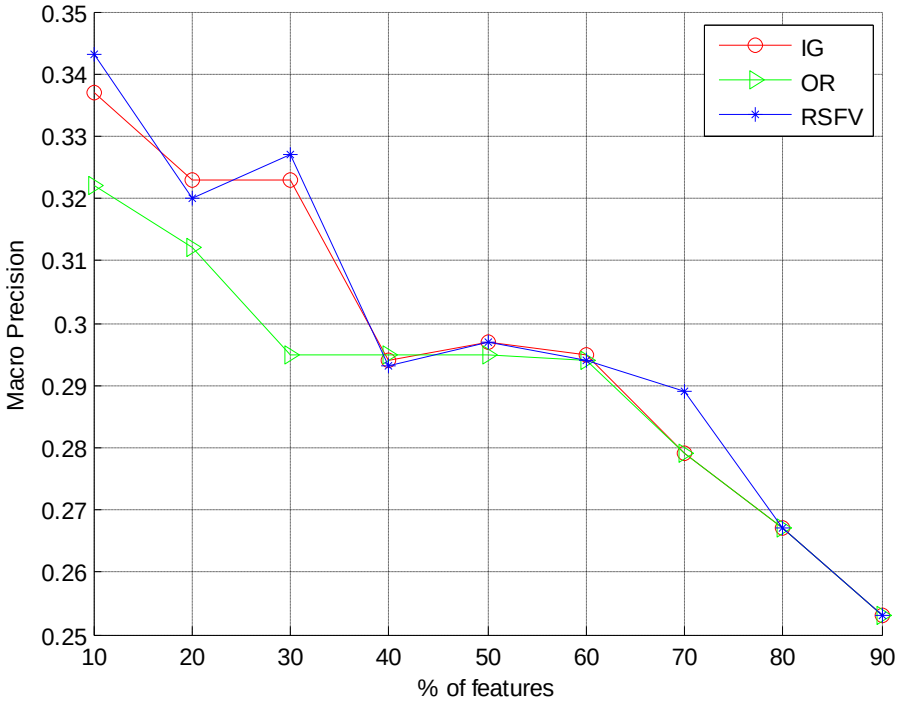


Figure 7.6 Macro-precision for Reuters with 10% – 90% features

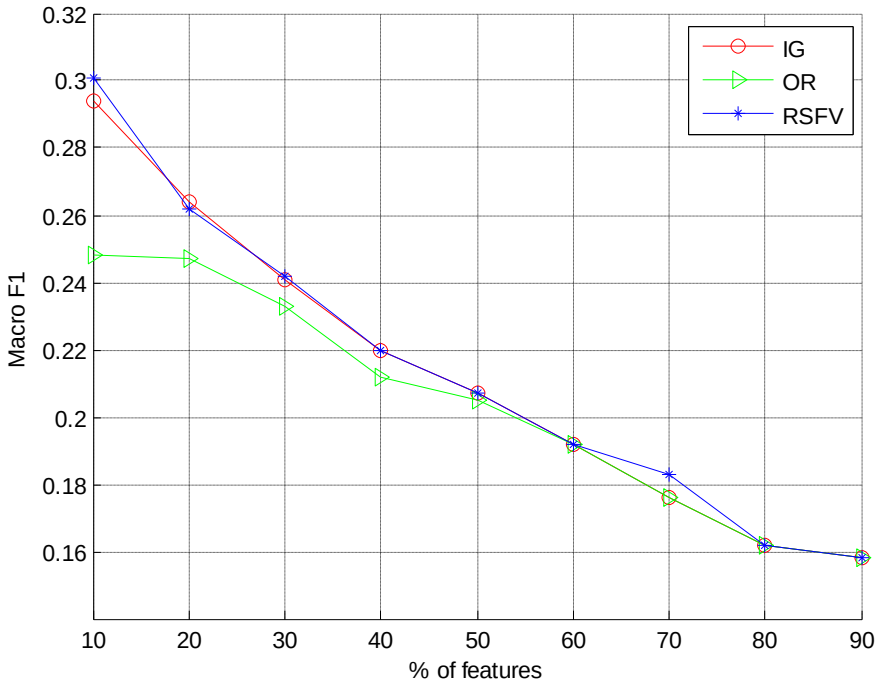


Figure 7.7 Macro-F_1 for Reuters with 10% – 90% features

As Figure 7.6 shows, when selecting top 70% of features, the macro-precision obtained by RSFV is slightly better (1%) than IG's. But for other cases, there is almost no difference between IG and RSFV. OR is still the worst among the three. We can say in general, RSFV achieves slightly better macro-precision than IG. Similarly to macro-recall, macro-precisions obtained by the three methods peak with the choice of top 10% of features.

As a combination of macro-recall and macro-precision, macro-F_1 in Figure 7.7 demonstrates the same trend as Figure 7.5. That is, macro-F_1 degrades as feature size enlarges. RSFV is better (up to 5.3%) than OR. It is considered equal to IG due to the small differences between them. From the three figures (Figure 7.5 – Figure 7.7), we can conclude that RSFV is much better than OR and equal or slightly better than IG with 10% – 90% of features in terms of macro-recall, precision and F_1.

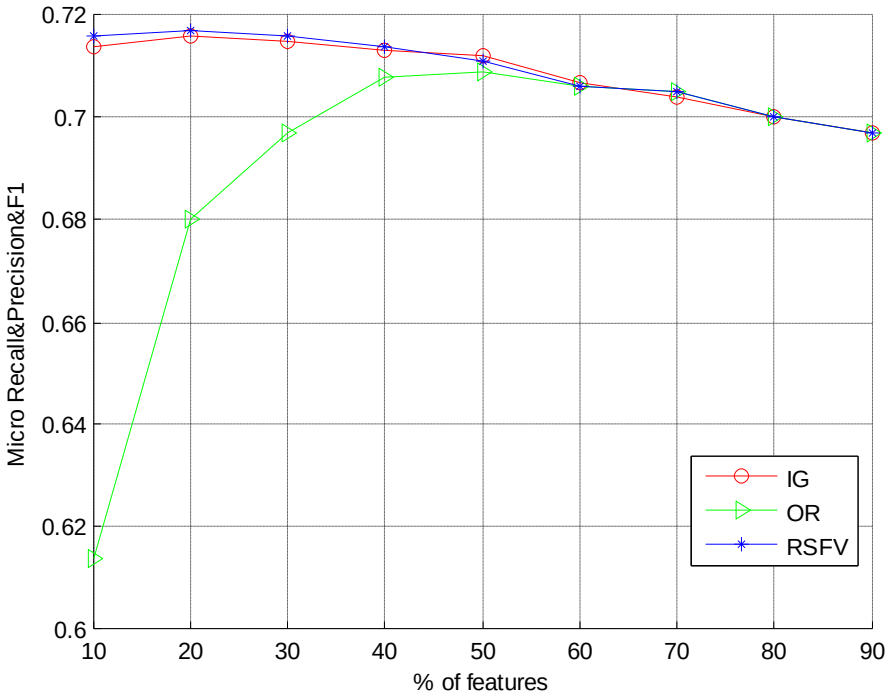


Figure 7.8 Micro- recall, precision, and F_1 for Reuters with 10% – 90% features

The comparisons of micro-based measures are shown in Figure 7.8. As we mentioned, because the naïve Bayes classifier assign each document to exact one category, micro-averaged recall, precision and F_1 have the same value. Again, the performance of RSFV and IG is very close, while OR is much worse than the others.

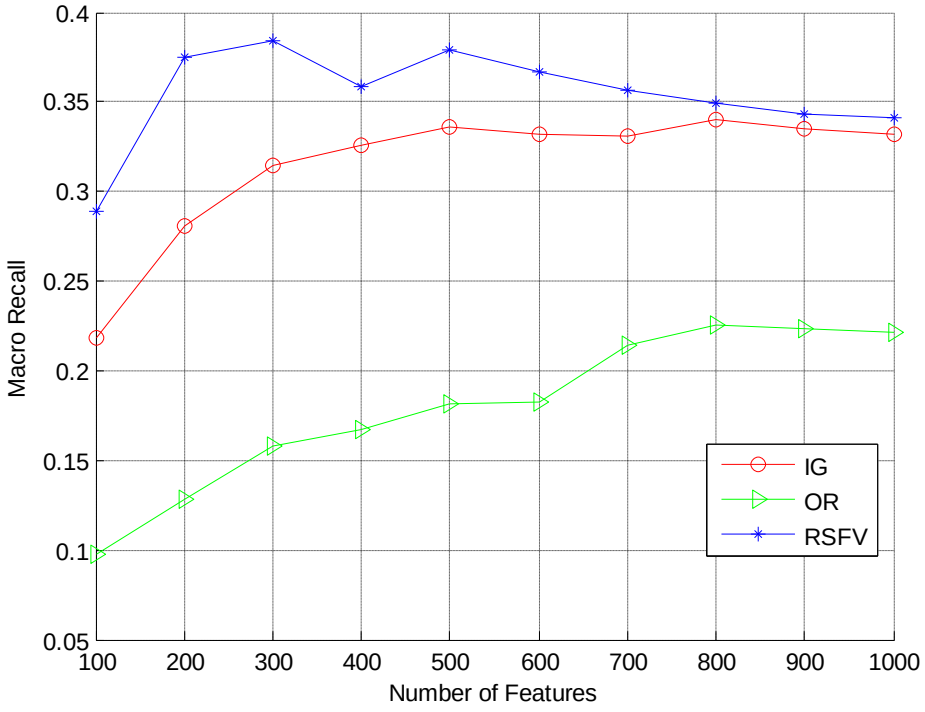


Figure 7.9 Macro-recall for Reuters with 100 – 1000 features

A common phenomenon of the four figures (Figure 7.5 – Figure 7.8) above is that performances degrades when the number of features increases (except OR in Figure 7.8). This suggests that the Reuters dataset may only need a small number of features for better classification results. Therefore, we further test and compare the performances of the methods under the condition that size of feature subsets is from one hundred to one thousand. We noticed that the differences of the performances between the three become more remarkable under this situation.

In Figure 7.9, it can be seen that RSFV is the best among them. OR's performance is much worse than the other two. From top 100 to 300 features selected, macro-recalls obtained by RSFV are much higher than those by IG with 7.1%, 9.4%, and 6.9% increase, respectively. RSFV provides a moderate (< 5%) increase when the feature size is from 400 to 700. For the rest cases, RSFV is slightly better than IG. The best macro-recall is accomplished by RSFV with the selection of top 300 features.

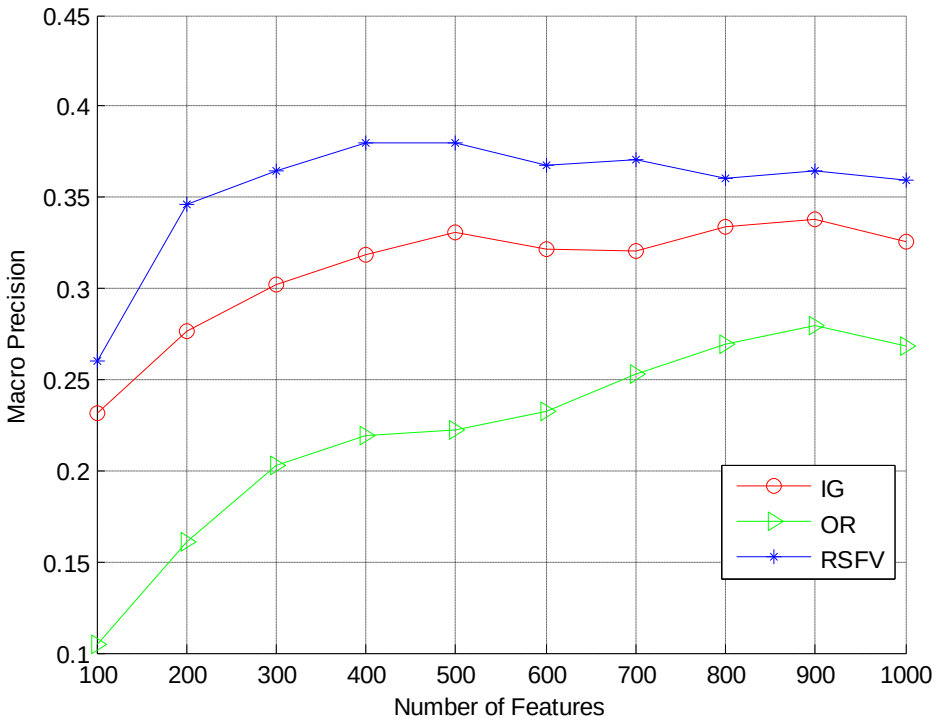


Figure 7.10 Macro-precision for Reuters with 100 – 1000 features

With respect to macro-precision shown in Figure 7.10, RSFV outperforms the others in every case too. RSFV outperforms the others in every case too. It achieves the highest macro-precision with top 400 or 500 features, which is different from the situation where best macro-recall is acquired. OR is still the worst among the three methods. When selecting features from 200 to

500 and 700, RSFV outperforms IG with remarkable increase (≥ 5%) up to 7%. In other cases, the increase ranges from 2.6% to 3.3%.

As for macro-F1, Figure 7.11 shows the similar observations that OR is not comparable with IG and RSFV and RSFV produces better results than IG and OR. The range of the improvement of macro-F1 obtained by RSFV over IG is from 1.5% to 6.8%.

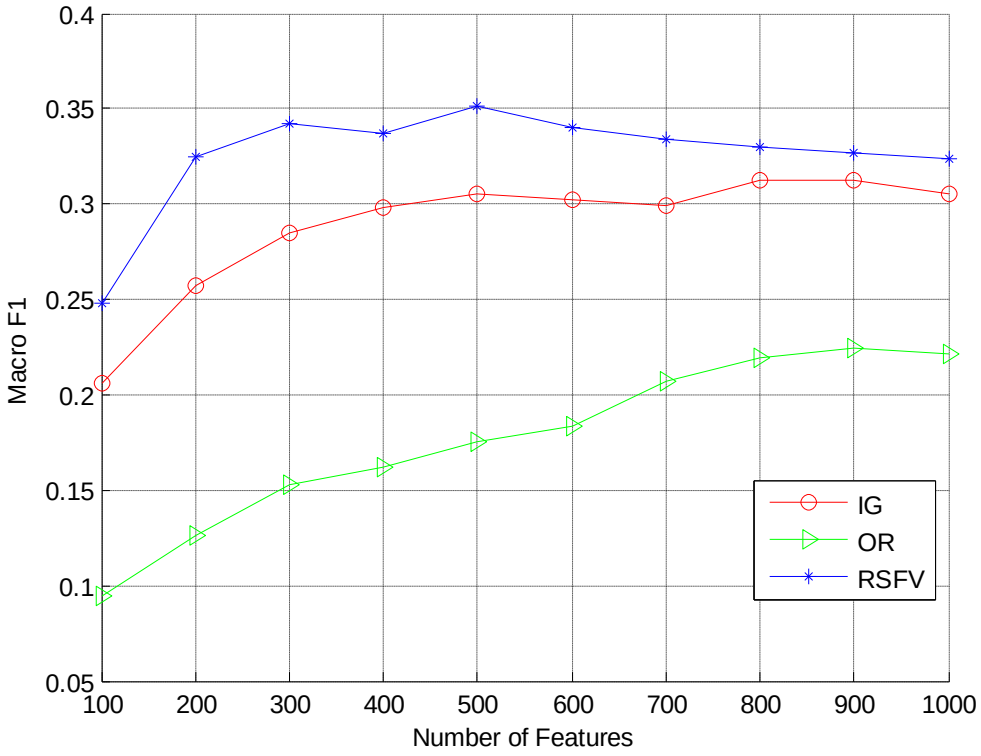


Figure 7.11 Macro-F_1 for Reuters with 100 – 1000 features

However, Figure 7.12 demonstrates IG performs better than the other two in terms of micro-recall, precision and F_1. The distinguishable differences between IG and RSFV lie in the first four points (100 - 400 features), while at the remaining points, the difference is about 1%. IG is 5.5% better than RSFV with 100 features selected. For other points (200 – 400 features), the increase of IG over RSFV is less than 3%.

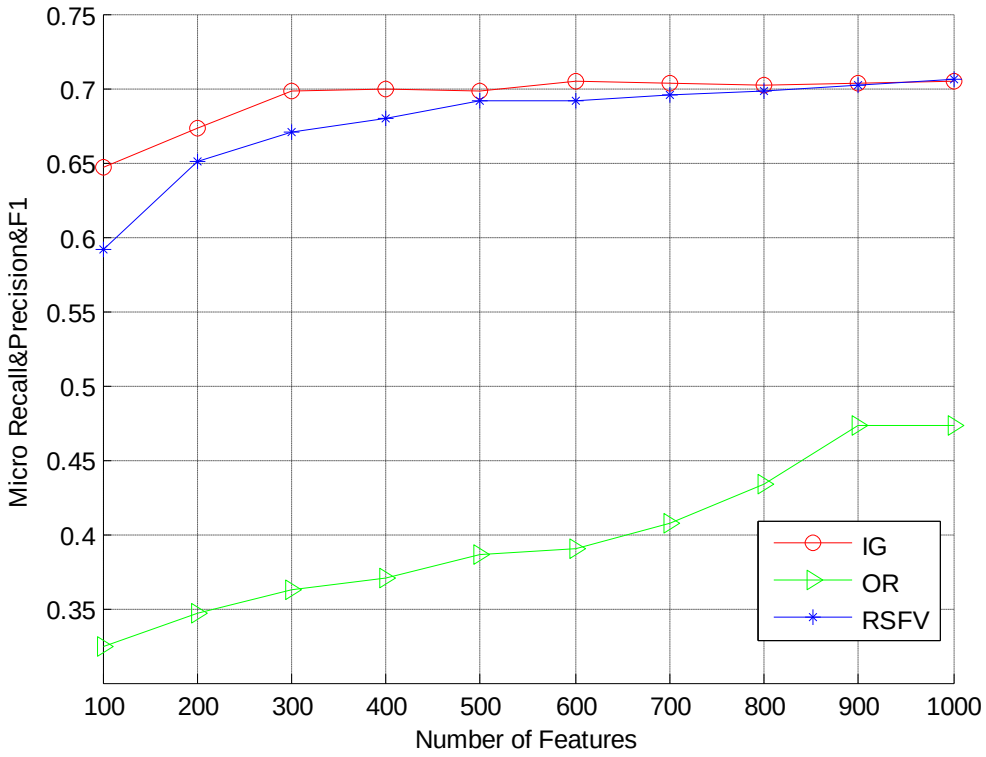


Figure 7.12 Micro-recall, precision, and F_1 for Reuters with 100 – 1000 features

Table 7.3 The Results of IG, OR and RSFV on Reuters with 10% – 90% features

Features	IG				OR				RSFV			
	*MA. Re.	*MA. Pre.	*MA. F_1	*MIs	MA. Re.	MA. Pre	MA. F_1	MIs	MA. Re.	MA. Pre.	MA. F_1	MIs
10%	0.305	0.337	0.294	0.714	0.241	0.322	0.248	0.614	0.305	0.343	0.301	0.716
20%	0.260	0.323	0.264	0.716	0.237	0.312	0.247	0.680	0.258	0.320	0.262	0.717
30%	0.231	0.323	0.241	0.715	0.222	0.295	0.233	0.697	0.230	0.327	0.242	0.716
40%	0.209	0.294	0.220	0.713	0.200	0.295	0.212	0.708	0.209	0.293	0.220	0.714
50%	0.195	0.297	0.207	0.712	0.192	0.295	0.205	0.709	0.195	0.297	0.207	0.711
60%	0.181	0.295	0.192	0.707	0.181	0.294	0.192	0.706	0.181	0.294	0.192	0.706
70%	0.167	0.279	0.176	0.704	0.168	0.279	0.176	0.705	0.173	0.289	0.183	0.705
80%	0.156	0.267	0.162	0.700	0.156	0.267	0.162	0.700	0.156	0.267	0.162	0.700
90%	0.151	0.253	0.158	0.697	0.151	0.253	0.158	0.697	0.151	0.253	0.158	0.697

*MA. Re.: Macro-Recall *MA. Pre.: Macro-Precision

*MA. F_1.: Macro-F_1

*MIs: Micro-Recall, Micro-precision, and Micro- F_1

Table 7.4 The Results of IG, OR and RSFV on Reuters with 100 – 1000 features

Features	IG				OR				RSFV			
	*MA. Re.	*MA. Pre.	*MA. F1	*MIs	MA. Re.	MA. Pre	MA. F1	MIs	MA. Re.	MA. Pre.	MA. F1	MIs
100	0.218	0.232	0.206	0.647	0.098	0.105	0.095	0.324	0.289	0.26	0.248	0.592
200	0.281	0.277	0.257	0.673	0.128	0.161	0.126	0.347	0.375	0.346	0.325	0.651
300	0.315	0.302	0.285	0.698	0.158	0.203	0.153	0.363	0.384	0.365	0.342	0.671
400	0.326	0.319	0.298	0.70	0.167	0.219	0.162	0.371	0.359	0.38	0.337	0.680
500	0.336	0.331	0.305	0.698	0.181	0.222	0.175	0.386	0.379	0.38	0.351	0.692
600	0.332	0.322	0.302	0.704	0.183	0.233	0.184	0.39	0.367	0.368	0.34	0.692
700	0.331	0.321	0.299	0.703	0.214	0.253	0.207	0.407	0.357	0.371	0.334	0.696
800	0.34	0.334	0.313	0.702	0.226	0.269	0.219	0.433	0.349	0.36	0.33	0.698
900	0.335	0.338	0.312	0.703	0.223	0.28	0.224	0.473	0.343	0.365	0.327	0.702
1000	0.332	0.326	0.305	0.704	0.221	0.268	0.221	0.473	0.341	0.359	0.324	0.706

*MA. Re.: Macro-Recall *MA. Pre.: Macro-Precision
*MA. F_1.: Macro-F_1
*MIs: Micro-Recall, Micro-precision, and Micro- F_1

We noticed that micro-recall, precision and F_1 is much higher than the macro-averaged recall, precision and F_1. Macro-averaged measures emphasize the performance of a classifier on categories with low generality, while micro-averaged measures focus on the overall performance of a classifier on all categories. The big difference between the results of macro-averaged and micro-averaged confirms that different categories of Reuters have very different generality. We provide the exact results of the three methods on Reuters text corpus, including macro-averaged and micro-averaged recall, precision, and F_1 in Table 7.3 and Table 7.4.

7.5 Summary

In this chapter, we proposed a simple but effective feature selection method called RSFV (Relation Strength and Frequency Strength) measure. The central hypothesis is that good features are highly correlated with the class and distribute most differently among all classes. The

stronger positive relation or negative relation indicates that a feature is more informative. Furthermore, a feature is more discriminative if its distribution is more variable across categories.

We compared our feature selection metric with two widely used metrics (IG and OR) in terms of different evaluation measures (i.e. micro- and macro- recall, precision and F_1) of classification. The experiments conducted on two benchmark corpora (Reuters-21578 and 20 Newsgroup) show that RSFV performs equal or better than IG and OR in most situations. They also suggest that the corpora with different properties can affect the performance of feature selection methods. No matter which feature subset we used (from 10% – 90% of features and from 100 – 1000 features), OR's performance is the worst in the experiment with Reuters-21578 data collection. On the other hand, RSFV obtains equal or better performance than IG in this experiment when selecting top 10% – 90% features. However, when we reduce the size of feature subsets to 100 – 1000 in the same experiment, RSFV shows remarkable improvements over IG and OR. Despite the poor performance in Reuters dataset, OR outperforms the other two in one or two cases in the other experiment with 20 Newsgroup. Nevertheless, it is worse than RSFV in the overall performance.

The two text collections have different distributions. In Reuters corpus, the distribution of documents across the categories is highly unbalanced. For example, some categories have few documents classified under them while others have thousands. However, in the 20 Newsgroup collection, articles are evenly distributed among 20 categories. In the experiment with Reuters corpus, our RSFV feature selection measure can achieve remarkable improvements over other methods. This is more impressive because the unbalanced data is more challenging for machine learning techniques.

Chapter 8

Conclusions and Future Work

As a technique to reduce dimenisonnality of data, feature selection is fundamental to improve the efficiency and effectiveness of machine leaning algorithms. The goal of this dissertation is to improve feature selection algorithms for machine learning areas.

8.1 Summary and Conclusions

The proliferation of feature selection techniques brought out the difficulty in choosing the best suitable feature selection algorithm for an application, which is resulted from the different feature selection criteria employed by different feature selection algorithms. In this dissertation, we proposed a hybrid genetic feature selection (HGFS) framework to solve the problem. The framework utilizes a feature pool, a genetic algorithm and an induction algorithm to combine multiple feature selection criteria. The feature pool collects valuable outcomes from multiple feature selection algorithms and/or human expertise and provides a good start point for the genetic algorithm to select feature subsets. The genetic algorithm calls a target induction algorithm to assess each candidate feature subset, which is a wrapper method. We first used a simple genetic algorithm whose goal is to maximize the classification accuracy. Then, we designed another genetic algorithm with a different goal. The second genetic algorithm considers

both the classification performance and the size of feature subsets. It aims to achieve a balance between the size of feature subsets and their classification performance.

We tested our framework in gene selection applications (i.e. colon cancer and prostate cancer datasets), which usually a large number of genes (features) but comparatively small number of data examples are involved. The tasks are to select discriminatory genes critical for cancer classification and diagnosis from DNA microarrays. Three traditional feature selection methods are used to form the feature pool in our framework, that is, SVM-RFE [102$], T-statistics, and an entropy-based feature selection method [16$]. For the induction algorithm, we choose SVM to evaluate feature subsets and validate the results. Both experiments show that our method can select feature subsets with better classification performance and/or smaller size than the each individual feature selection algorithm does. The combination of different feature selection criteria not only improves the classification performance of the feature subsets selected, but also is capable of finding good feature subsets for various applications. In addition, our framework makes good understanding of application domains and the technical details of the algorithms unnecessary, which is a good choice for different applications.

Text categorization is a booming application domain that classifies documents into predefined categories. It typically includes hundreds to thousands of features and data examples, which makes wrapper methods computational costly. Therefore, we proposed a simple filter approach for text categorization applications. The proposed feature selection metric called Relation Strength and Frequency Variance (RSFV) assesses features from two perspectives: (1) the strength of positive relation or negative relation between a feature and a class and (2) the degree of variability of distributions of a feature among all categories. We think informative features are those that are highly correlated with the class and whose distributions vary most

among all classes. The positive relation of a feature with a class is measured by the co-occurrences and co-non-occurrences of the feature and the class, while the negative relation is measured by the non-co-occurrences of the two. On the other hand, we use variance of the term frequency of a feature across all categories to measure the degree of variability of its distributions. Feature selection is based on the ranking list of features generated by RSFV scoring function.

Experiments are conducted on two standard text corpora, Reuters-21578 and 20 Newsgroup. RSFV is compared with two widely used metrics (IG and OR) in terms of different evaluation measures (i.e. micro- and macro- recall, precision and F_1) of classification for different purposes. The feature subset size is either proportional to the original feature set or a predefined number. The experimental results reveal that RSFV obtains equal performance or outperforms other traditional methods in many situations. In the experiment with a balanced text collection (i.e. 20 Newsgroup), although OR can achieve best performance at one or two points, RSFV is the best in terms of overall performances among the three methods. The second experiment is conducted on a highly unbalanced text collection (i.e. Reuters-21578), which is a more difficult task for machine learning. We can see that RSFV shows remarkable improvement ($\geq$ 5%) over other feature selection methods.

8.2 Future Work

A limitation of our hybrid feature selection framework is that it requires much computation time because the genetic algorithms repeatedly call the induction algorithm for evaluation of

feature subsets, a common drawback of wrapper method. In the future, we can speed up our method by parallelizing the genetic algorithms.

In the experiments of text categorization, we chose a naïve Bayes classifier to classify documents. Naïve Bayes classifiers are efficient, but their performances may not be as good as others. We would like to explore how RSFV performs with different classifiers, such as SVM.

Currently we use global feature selection for text categorization. That is, a fixed feature subset is selected for the classification of all categories. This may not generate the best performance for each category. On the other hand, local feature selection chooses a different feature subset for the classification of each category, which is supposed to improve the classification performance for each category. In the future, we can apply RSFV for local feature selection to see how it works.

The bag-of-words model we adopted assumes that features are independent of each other. In addition, it does not consider the positions of words. However, the correlations and positions of words can carry extra meaningful information. For example, a word phrase provides more information than each individual word in the phrase. In the future, we can deal with these types of information.

Currently we used simple linguistic preprocessing in the experiments. We will explore if more preprocessing, such as misspelling corrections, removal of terms with low document frequency, can improve the performance of text categorization.

We used precision, recall and F1-measure to evaluate the performance of text categorization. The classification performance can be evaluated in another point of view. For example, we can measure the areas under Precision-Recall curves or under ROC (Receiver Operator Characteristics) curves for the evaluation. We will address this in the future.

8.3 List of Publications

[1] F. Tan, X. Fu, H. Wang, Y. Zhang and A. Bourgeois. A Hybrid Feature Selection Approach for Microarray Gene Expression Data. Proc. of IWBRA2006, May 28-31, 2006

[2] F. Tan, X. Fu, Y. Zhang and A. Bourgeois. Improving Feature Subset Selection Using Genetic Algorithm for Microarray Gene Expression Data. Proc. of CEC2006, Vancouver, July 16-21, 2006

[3] F. Tan, X. Fu, Y. Zhang and A. Bourgeois. A Genetic Algorithm-based Feature Subset Selection. Soft Computing - A Fusion of Foundations, Methodologies and Applications, 2007

[4] F. Tan, A.G. Bourgeois and E. Cho. A Simple Feature Selection Metric for Text Categorization. Submitted to SIAM International Conference on Data Mining (SDM08), 2008

[5] E. Cho, A. G. Bourgeois, and F. Tan. An FPGA Design to Achieve Fast and Accurate Results for Molecular Dynamics Simulations. Proc. of ISPA-2007, Niagara Falls, Canada, August 29 – 31, pp. 256-267, 2007

[6] H. Wang, F. Tan, A. Sabnis, X. Fu, P. Volarath and R. Harrison. Pluggable Application Server Framework. EMBC06. New York, USA. 2006

[7] X. Fu, F. Tan, H. Wang, Y. Zhang and R. Harrison. Feature Similarity Based Redundancy Reduction for Gene Selection. DMIN'06, Las Vegas, Nevada, USA, 2006

Bibliography

[1] Aas, L. and Eikvil, L. Text categorization: A survey. Raport NR 941, Norwegian Computing Center, 1999

[2] Alon, U. et al. Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays, PNAS, 96:6745-6750, 1999

[3] Baker, L. D. and McCallum, A. Distributional clustering of words for text classification. In SIGIR '98: Proc. the 21st Annual International ACM SIGIR, pp. 96-103. 1998

[4] Blum, A. L. and Langley, P. Selection of relevant features and examples in machine learning. Artificial Intelligence, pages 245-271, 1997

[5] Breiman, L. Random Forest. Technical Report, Stat.Dept. UCB. 2001

[6] Brill, F. Z., Brown, D. E. and Martin, W. N. Fast Genetic Selection of Features for Neural Network Classifiers. IEEE Trans. Neural Networks, vol. 3, no. 2, pp. 324-328, 1992

[7] Buckley, C., Salton, G., Allan, J. and Singhal, A. Automatic query expansion using SMART: TREC 3. In Proc. 3rd Text Retrieval Conference, NIST, 1994

[8] Burges, C. J. C. A tutorial on support vector machines for pattern recognition. Data mining and Knowledge Discovery, 2(2), 121-167, 1998

[9] Chuang, H.Y. et al. Identifying Significant Genes from Microarray Data. Fourth IEEE Symposium on Bioinformatics and Bioengineering (BIBE'04), p. 358, 2004

[10] Chuang, W. T. et al. A Fast Algorithm for Hierarchical Text Classification. Data Warehousing and Knowledge Discovery, pp. 409-418, 2000

[11] Cohen, W. W. Learning to classify English text with ILP methods. In Advances in Inductive Logic Programming, L.De Raedt, ed. IOS Press, Amsterdam, The Netherlands, 124-143. 1995

[12] Combarro, E. F., Montanes, E. Diaz, I. Ranilla, J. and R. Mones. Introducing a Family of Linear Measures for Feature Selection in Text Categorization. IEEE Trans. Knowl. Data Eng, 17(9), pp. 1223-1232, 2005

[13] Cover, T. M. and Hart, P. E. Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13:21-27, 1967

[14] Das, S. Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection. Proc. 18th Int'l Conf. Machine Learning, pp. 74-81, 2001

[15] Dash, M. and Liu, H. Feature selection for classification. International Journal of Intelligent Data Analysis, 1(3), 1997

[16] Dash, M. and Liu, H. Handling large unsupervised data via dimensionality reduction. ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. 1999

[17] Dash, M., Choi, K., Scheuermann, P. and Liu, H. Feature Selection for Clustering - A Filter Solution. in Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02), pp. 115-122, 2002

[18] Doak, J. An Evaluation of Feature Selection Methods and Their Application to Computer Security. Technical report, Univ. of California at Davis, Dept. Computer Science, 1992

[19] Drucker, H. Vapnik, V. and Wu, D. Support vector machines for spam categorization. IEEE Transactions on Neural Networks, 10(5), pp. 1048-1054, 1999

[20] Dumais, S., Platt, J., Heckerman, D. and Sahami, M. Inductive learning algorithms and representations for text categorization. Proc. 7th Int'l Conf. Information and knowledge management, pp. 148-155, 1998

[21] Dy, J. G. and Brodley, C. E. Feature Selection for Unsupervised Learning. The Journal of Machine Learning Research, vol. 5, pp.845-889, Aug. 2004

[22] Forman, G. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. Journal of Machine Learning Research, 3, 1289-1305, 2003

[23] Frakes, W. B. Stemming algorithms. Information Retrieval: Data Structures and Algorithms, eds. W.B. Frakes & R. Baeza-Yates, Prentice Hall: Englewood Cliffs, US, pp. 131-160, 1992

[24] Furey, T., Cristianini, N. and Duffy, N. Bednarski, D.M. Schummer and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics, 16:906–914, 2000

[25] Galavotti, L. Sebastiani, F. and Simi, M. Experiments on the use of feature selection and negative evidence in automated text categorization. In: Proceedings of ECDL-00, 4th European conference on research and Advanced Technology for Digital Libraries, Lisbon, Portugal, pp. 59-68, 2000

[26] Golub, T.R. et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science, 286, 1999

[27] Guyon, I. and Elisseeff, A. An Introduction to Variable and Feature Selection (Kernel Machines Section). JMLR, 3: 1157-1182, 2003

[28] Hall, M. A. Correlation-Based Feature Selection for Discrete and Numeric Class Machine Learning. Proc. 17th Int'l Conf. Machine Learning, pp. 359-366, 2000

[29] Hall, M. A. Correlation-based Feature Selection for Machine Learning. Ph.D. thesis, Department of Computer Science, Waikato University, New Zealand, 1999

[30] Hastie, T., Tibshirani, R. and Friedman, J. The Elements of Statistical Learning. Springer series in statistics. Springer, New York, 2001

[31] Holland, J. Adaptation in Nature and Artificial Systems. MIT Press, 1992.

[32] Hsu, D. F., Shapiro, J. and Taksa, I. Methods of Data Fusion in Information Retreival: Rank vs. Score combination. DIMACS Technical Report 58, 2002

[33] Jain, A. K. Murthy, M. N. and Flynn, P. J. Data clustering: A review. ACM Computing Surveys, 31(3), 1999

[34] Jirapech-Umpai, T. and Aitken, S. Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes. BMC Bioinformatics, 6:148, 2005

[35] Joachims, T. A probabilistic analysis of the Rochhio algorithm with TFIDF for text categorization. In Machine Learning: Proceedings of the Fourteenth International Conference, pp. 143-151, 1997

[36] Joachims, T. Text categorization with support vector machines: learning with many relevant features. Proc. of ECML-98, 10th European Conference on Machine Learning (Chemnitz, Germany), 137-142. 1998

[37] John, G. H., Kohavi, R. and Pfleger, K. Irrelevant features and the subset selection problem. In: Proc. the Eleventh International Conference on Machine Learning, 121-129, 1994

[38] Jolliffe, I. T. Principal Component Analysis. New York: Spriger Verlag, 1986.

[39] Kent Ridge Biomedical Data Set Repository. Available at http://sdmc.lit.org.sg /GEDatasets/Datasets.html

[40] Kira, K. and Rendell, L. A. The feature selection problem: Traditional methods and a new algorithm. In: Proceedings of Ninth National Conference on Artificial Intelligence, 129-134, 1992

[41] Kohavi, R. and John, G. H. Wrappers for Feature Subset Selection. Artificial Intelligence, vol. 97, nos. 1-2, pp. 273-324, 1997

[42] Kohonen, T. Self-Organizing Maps. Series in Information Sciences, Vol. 30. Springer, Heidelberg. Second ed. 1997

[43] Koller, D. and Sahami, M. Toward optimal feature selection. In: Proceedings of International Conference on Machine Learning, 1996

[44] Kudo, M. and Sklansky, J. Comparison of algorithms that select features for pattern classifiers. Pattern Recognition 33, pp. 25-41, 2000

[45] Langley, P., Iba, W. and Thompson, K. An analysis of Bayesian classifier. Proc. Tenth National Conference on Artificial Intelligence, 223-228, 1992

[46] Lawrence, J. Introduction to Neural Networks. California Scientific Software Press. ISBN 1-883157-00-5, 1994

[47] LeCun, Y., Denker, J. S. and Solla, S. A. Optimum Brain Damage. Advances in Neural Information Processing Systems II, D.S.Touretzky, Ed. Mateo, CA: Morgan Kaufmann, 1990

[48] Lewis, D. Available at http://www.daviddlewis.com/resources/testcollections /reuters21578/

[49] Lewis, D. D. Feature Selection and Feature Extraction for Text Categorization. Proc. Speech and Natural Language Workshop, pp. 212-217, 1992

[50] Lewis, D. D. Representation and Learning in Information Retrieval, Ph. D. thesis. Department of computer Science, University of Massachusetts, Amherst, MA. 1992

[51] Liu, H. and Motoda, H. Feature Selection for Knowledge Discovery and Data Mining. Boston: Kluwer Academic, 1998

[52] Liu, H. and Setiono, R. Chi2: Feature selection and discretization of numeric Attributes. Proc. IEEE 7th International Conference on Tools with artificial Intelligence, 338-391, 1995

[53] Liu, H. and Yu, L. Toward Integrating Feature Selection Algorithms for Classification and Clustering. IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 4, pp. 491-502, Apr., 2005

[54] Liu, H. et al. Evolving Feature Selection. Intelligent Systems. IEEE Volume 20, Issue 6, Nov.-Dec. Page(s):64-76, 2005

[55] Liu, H. Li, J. and Wong, L. A comparative study on feature selection and classification methods using gene expression profiles and proteomic pattern. Genomic Informatics, 13, 51-60, 2002

[56] Liu, X. A. Krishnan and A. Mondry. An Entropy-based gene selection method for cancer classification using microarray data. BMC Bioinformatics, 6: 76, 2005

[57] Liu, Y. A comparative study on feature selection methods for drug discovery. Journal of Chemical Information and Computer Sciences 44(5): 1823-1828, 2004

[58] Mao, Y., Zhou, X., Pi, D., Sun, Y. and Wong, S. T. C. Multiclass Cancer Classification by Using Fuzzy Support Vector Machine and Binary Decision Tree with Gene Selection. J Biomed Biotechnol, 2005(2): 160–171, 2005

[59] Martinez, A. M. and Kak, A. C. PCA versus LDA. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23. 228 - 233. 2001

[60] McCallum, A. and Nigam, K. A Comparison of Event Models for Naive Bayes Text Classification. In AAAI/ICML-98 Workshop on Learning for Text Categorization, pp. 41-48. Technical Report WS-98-05. AAAI Press. 1998

[61] McCallum, A. Bow toolkit. Available at http://www.cs.cmu.edu/~mccallum/bow/

[62] Mitchell, T. Machine Learning. McGraw Hill, 1997

[63] Mladenic, D. and Grobelnik, M. Feature selection for unbalanced class distribution and Naive Bayes. Proc. 16th Int'l Conf. Machine Learning(ICML-99), pp. 258-267, 1999

[64] Mladenic, D. Machine learning on non-homogeneous, distributed text data. PhD thesis, University of Ljubljana, Slovenia, October, 1998

[65] Montanes, E., Díaz, I., Ranilla, J., Combarro, E. F. and Fernandez, J. Scoring and Selecting Terms for Text Categorization. IEEE Intelligent Systems, 20(3), pp. 40-47, 2005

[66] Narendra, P. M. and Fukunaga, K. A Branch and Bound Algorithm for Feature Subset Selection. IEEE Trans. Computer, vol. 26, no. 9, pp. 917-922, Sept. 1977

[67] Noble, W. S. Support vector machine applications in computational biology. Kernel Methods in Computational Biology. B. Schoelkopf, KTsuda and J.-P. Vert, ed. MIT Press, 71-92, 2004

[68] Oh, I. S., Lee, J. S. and Moon, B. R. Hybrid genetic algorithms for feature selection. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 26, Issue 11, pp. 1424- 1437, Nov. 2004

[69] Olsson, J. S. and Oard, D. W. Combining feature selectors for text classification. Proc. the 15th ACM international conference on Information and knowledge management, pp. 798-799, 2006

[70] Pekar, V., Krkoska, M. and Staab, S. Feature weighting for co-occurrence-based classification of words. Proc. 20th Int'l Conf. Computational Linguistics, article No. 799, 2004

[71] Peng, H. C., Long, F. H. and Ding, C. Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 27, no. 8, pp. 1226–1238, 2005

[72] Piatetsky-Shapiro, G. and Tamayo, P. Microarray Data Mining: Facing the Challenges, SIGKDD Explorations Special Issue on Microarray Data Mining, pp. 1-5, Volume 5, Issue 2, December 2003

[73] Porter, M. F. An algorithm for suffix stripping. In Program, 14(3), 1980

[74] Press, W. H., Flannery, B. P., Teukolsky, S. A. and Vetterling, W. T. Numerical recipes in C. Cambridge University Press, Cambridge. 1988

[75] Quinlan, J.R. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993

[76] Raymer, M. L., Punch, W. F., Goodman, E. D. Kuhn, L. A. and Jain, A. K. Dimensionality Reduction Using Genetic Algorithms. IEEE Trans. Evolutionary Computation, vol. 4, no. 2, pp. 164-171, 2000

[77] Rennie, J. ifile: An application of machine learning to e-mail filtering. Proc. KDD Workshop on Text Mining, 2000

[78] Rich, E. and Knight, K. Artificial Intelligence. McGraw-Hill, 1991

[79] Rijsbergen, C. J. V. Information Retrieval, 2nd ed. Butterworths, London, UK. Available at http://www.dcs.gla.ac.uk/Keith. 1979

[80] Rogati, M. and Yang, Y. High-performing feature selection for text classification. CIKM: 659-661, 2002

[81] Roweis, S. T. and Saul, L. K. Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science, Dec. 22; 290(5500):2323-6. 2000

[82] Russell, J. S. and Peter, N. Artificial Intelligence: A Modern Approach (2nd ed.). Upper Saddle River, NJ: Prentice Hall, pp. 111-114, ISBN 0-13-790395-2. 2003

[83] Sahami, M., Dumais, S., Heckerman, D. and Horvitz, E. A Bayesian approach to filtering junk e-mail. In Proceedings of AAAI-98 Workshop on Learning for Text Categorization, pp. 55-62, Madison, Wisconsin, 1998

[84] Salton, G. and Buckley, C. Term-weighting approaches in automatic text retrieval. Inform. Process. Man. 24, 5, 513-523. 1988. Also reprinted in Sparck Jones and Willett, pp. 323–328. 1997

[85] Salton, G. and McGill, M. J. An Introduction to Modern Information Retrieval. McGraw-Hill, 1983

[86] Schölkopf, B., Guyon, I. and Weston, J. Statistical Learning and Kernel Methods in Bioinformatics. Artificial Intelligence and Heuristic Methods in Bioinformatics 183, (Eds.) P. Frasconi und R. Shamir, IOS Press, Amsterdam, The Netherlands, 1-21, 2003

[87] Sebastiani, F. Machine learning in automated text categorization. ACM Computing Surveys, 34(1), 1-47. 2002

[88] Sebastiani, F. Text Categorization. Encyclopedia of Database Technologies and Applications, pp. 683-687, 2005

[89] Singh, D. et al. Gene Expression Correlates of Clinical Prostate Cancer Behavior. Cancer Cell, 1:203-209, March, 2002

[90] Singh, D. Gene Expression Correlates of Clinical Prostate Cancer Behavior. Cancer Cell, 1:203-209, March, 2002

[91] Somorjai, R. L., Dolenko, B. and Baumgartner, R. Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions. Bioinformatics, 19(12):1484-91, 2003

[92] Space Physics Group. Applied Physics Laboratory, Johns Hopkins University. Johns Hopkins Road, Laurel, MD 20723

[93] Tan, F., Bourgeois, A. G. and Cho, E. A Simple Feature Selection Metric for Text Categorization. Submitted to SIAM International Conference on Data Mining (SDM08), 2008

[94] Tan, F., Fu, X., Wang, H., Zhang, Y. and Bourgeois, A. G. A Hybrid Feature Selection Approach for Microarray Gene Expression Data. Proc. of IWBRA2006, May 28-31, 2006

[95] Tan, F., Fu, X., Zhang, Y. and Bourgeois, A. G. A Genetic Algorithm-based Feature Subset Selection. Soft Computing - A Fusion of Foundations, Methodologies and Applications, 2007

[96] Tan, F., Fu, X., Zhang, Y. and Bourgeois, A. G. Improving Feature Subset Selection Using Genetic Algorithm for Microarray Gene Expression Data. Proc. of CEC2006, Vancouver, July 16-21, 2006

[97] The Apache SpamAssassin Project. Available at http://www.spamassassin.org

[98] Vafaie, H. and Jong, K. D. Genetic Algorithms as a Tool for Feature Selection in Machine Learning. 4th Int'l Conf. Tools with Artificial Intelligence. IEEE Computer Society Press. pp. 200-203. 1992

[99] Vapnik, V. Estimation of Dependencies Based on Empirical Data. New. York: Springer-Verlag, 1982

[100] Vapnik, V. Statistical Learning Theory. New York, John Wiley and Sons, 1998.

[101] Wang, G. and Lochovsky, F. H. Feature selection with conditional mutual information maximin in text categorization. Proceedings of the thirteenth ACM international conference on Information and knowledge management, pp. 342-349, 2004

[102] Weston, I.J., Barnhill, S. and Vapnik, V. Gene selection for cancer classification using Support Vector Machines. Machine Learning, Vol. 46, No. 1-3, pp. 389-422, 2002

[103] Weston, J. et al. Feature selection for SVMs. Advances in Neural Information Processing Systems 13, 2000

[104] Wibowo, W. and Williams, H. E. Simple and accurate feature selection for hierarchical categorization, Proceedings of the 2002 ACM symposium on Document engineering, pp: 111-118, 2002

[105] Xing, E., Jordan, M. and Karp, R. Feature Selection for High-Dimensional Genomic Microarray Data. Proc. 15th Int'l Conf. Machine Learning, pp. 601-608, 2001

[106] Yang, J. and Honavar, V. Feature subset selection using a genetic algorithm. IEEE Intelligent Systems, 13:44-49, 1998

[107] Yang, Y. and Liu, X. A re-examination of text categorization methods. In Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval (Berkeley, CA), 42-49. 1999

[108] Yang, Y. and Pedersen, J. O. A comparative study on feature selection in text categorization. Proceedings of ICML-97, 14th International Conference on Machine Learning, ed. D.H. Fisher, Morgan Kaufmann Publishers, San Francisco, US: Nashville, US, pp. 412–420, 1997

[109] Yu, L. and Liu, H. Efficiently Handling Feature Redundancy in High-Dimensional Data. Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD 03), ACM Press, pp. 685-690, 2003

[110] Yu, L. and Liu, H. Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. Proc. 20th Int'l Conf. Machine Learning, pp. 856-863, 2003