CHAPTER II

BASIC BACKGROUND AND RELATED TOPICS

The idea of using multiple models to a given pattern recognition problem has been studied in the past and continued with great successes today. In Sections 2.1 and 2.3, we review these prior multiple model algorithms which we intend to modify them to improve their performance with our theory. Section 2.2 briefly describes various methods for construction MCS. Sections 2.4 is devoted to a classical theory and two new formal definitions in which we are inspired by them for our computational classification models in the following chapters.

2.1 Introduction

There are many ways to use more than one classifier in a single classification problem – continuum of combination schemes. One basic approach is rather conservative, in which we refuse to take any specific notice of models. We can also consider this approach as *parallel approach*. The second approach is considered as *sequential approach*, in which each model is domain- specific trained. The advantage of the parallel combination is that it is easier than figuring out what is good in each classifier. This is from the fact that the training data and training process are usually the limiting factors. This is why many researchers still conduct their ongoing research in MCS toward the parallel approach.

A *multiple classifier system* consists of a set of classifiers and a decision combination function. The construction strategies of parallel approach usually fall into two categories: (1) assume a fixed decision combination function, *generate* a set of mutually complementary, generic classifiers that can be combined to achieve optimal accuracy; and (2) assume a given, fixed set of carefully designed and highly specialized classifiers, attempt to find an *optimal combination* of their predictions (decisions).

We will refer to combination strategies of the first kind as *coverage optimization* methods and the second kind as *prediction optimization* methods, where various techniques purposed for both strategies are summarized in Tables 2.1 and 2.2 respectively. It is also possible to apply the decision optimization methods to classifiers generated with the aim of coverage optimization.

In this dissertation, we mainly focus in combining strong learners approach rather than weak. The effectiveness of the combining strong learners approach is come from further reducing variance, while the combining weak learned approach is aimed at reducing both bias and variance. In combining strong learners, both coverage and prediction optimization strategies should be chosen carefully with the aim at reducing variance; it is in this context that most of the techniques presented in this dissertation are studied rigorously.

Table 2.1 Coverage optimization methods.

Method	Training mechanism for introducing complementariness	Physical level
noise injection	inject noise to the training set	input
stacking	train classifiers by nonoverlapping subsamples of training set	input distribution $\mathcal{D}$
bagging	resample the training set by bootstrap replicates	input distribution $\mathcal{D}$
boosting	resample the training set by weights evolving with accuracy	input distribution $\mathcal{D}$
random subspace	project training set to random chosen subspaces	input representation
stochastic discrimination	generate random kernels to measure coverage of training set	classifier
perturbation	vary initial conditions or parameters of training process	classifier
error correction output coding	force training on partial decision boundaries	output
multiple feature sets	train classifiers using different feature subsets	input representation

Table 2.2 Prediction optimization methods.

Trainable	binary or one of N decisions	ranked lists of classes	continuous prob. estimates or belief scores
No	majority	order statistics	sum, product rules
Yes	weighted vote	logistic regression	Bayes rules

2.2 Construction Approaches for Multiple Classifier Systems

In practice, many methods for constructing multiple classifier systems (MCS) [22, 23] have been developed, which we can group them into several categories based around five main ideas: (1) Bayesian voting by ensembling all hypotheses h in $\mathcal{H}$, each weighted by its posterior probability $P(h/\mathbf{x})$; (2) resampling training examples in order to find optimal training sample representativeness; (3) manipulating the output targets by training on partial coverage decision boundaries; (4) injecting randomness by perturbation sampling or using randomized classification algorithms; and (5) manipulating the input features. With corresponding to the above categorization sequence. these techniques comprise Markov Chain Monte Carlo (MCMC) [24], Adaboost [12] and its variants, error correction output coding [4], perturbation and random subspace [25], and input decimation methods [26], respectively.

After some of the most remarkable recent studies in MCS, some authors [27] argue that the important of interpretability has been marginalized in the design of these algorithms, and put behind the need to devise classifiers with strong classification power. Some authors have pointed out the interest to define new definitions [8, 28] leading to several new interpretations of MCS, but they are limited to the use of weak learners. They are not concerning with the quantification and qualification of MCS in case of using learners with strong classification power.

Empirically, we found that in some cases using strong learners in MCS is more efficient than using weak learners both in terms of computational complexity and generalization. We have learned that the choice of learners for MCS, which can be described using accuracy–simplicity space, is data dependent. Thus, putting behind the need to devise classifiers with strong classification power should not be indispensable in general. This is why we review some MCS and also their new formal definitions necessary for understanding our new coverage and combining algorithms.

2.3 A Review of Some Multiple Classifier Systems

In this section, we review the multiple classifier systems used in our study, i.e., Adaboost, ECOC, and random subspace methods. Various kinds of pattern classifiers are studied for using with the above systems, e.g., Decision Trees, neural networks, support vector machine, linear discriminant analysis, and k-nearest neighbor (k-NN). The useful information on multiple classifier systems and pattern classifiers in general can be found in [23] and [29–33], respectively.

2.3.1 Bagging

Bagging [34] is a general method of combining classifiers that can be applied to any base method. In Bagging, n data sets are created by sampling patterns with replacement

from the original training set. Each of the n data sets has the same number of patterns as the original training set. This sampling method (bootstrap sampling) results in each pattern appearing in a given data set with approximately 63.2 % probability and for each data set, we train a base classifier to distinguish all classes. After training n classifiers, we combine their output using simple voting. It should be noted that Bagging is not effective with nearest neighbor classifiers.

2.3.2 Adaboost

AdaBoost is a boosting algorithm, running a given weak learner several times on slightly altered training data, and combining the hypotheses to one final hypothesis, in order to achieve higher accuracy than the weak learner's hypothesis would have. The main idea of AdaBoost is to assign each example of the given training set a weight. At the beginning all weights are equal, but in every round the weak learner returns a hypothesis. and the weights of all examples classified wrong by that hypothesis are increased. That way the weak learner is forced to focus on the difficult examples of the training set. The final hypothesis is a combination of the hypotheses of all rounds, namely a weighted majority vote, where hypotheses with lower classification error have higher weight.

In detail: Given

- A set $E = \{(x_1, y_1), ..., (x_n, y_n)\}$ of classified examples, where $x_i \in X$ and $y_i \in Y$, for $i = 1, ..., n$. Here we assume $Y = -1, +1$, e.g. instances that are not covered by a concept to be learned have label -1 and the ones covered have label +1.

- A weak learning algorithm deals with weighted example sets. Such a learning algorithm reads an example set E and a distribution D. In the most simple case, where all hypotheses that can be output are functions $Y = -1, +1$, the algorithm tries to find a hypothesis h with minimal probability of misclassification, given that an example is drawn from X with respect to D. The case of other possible hypotheses can be addressed by using more complex error measures.

We have shown the Adaboost algorithm in Figure 2.3.2.

2.3.3 Error Correcting Output Codes

Error correcting output codes (ECOC) proposed by Dietterich and Bakiri [4] is a method to creating and combining classifiers by output decomposition from multi-class problems into a series of two-class problems.

In ECOC, we assign a codeword or binary string for each class. Each bit position in the string corresponds to a two-class problem (dichotomy of classes) generated by decomposing the classes into two disjoint sets. For example, given four classes: A, B, C, and D, then the dichotomies of classes could correspond to separating {A} from {BCD}, {B} from

{ACD}, {AD} from {BC}, and so on. For each position we train a classifier which learn to discriminate the two sets.

To classify an unknown pattern, the pattern is presented to all of the classifiers. The sequence of outputs is a new string which corresponds to the unknown pattern. When the dichotomies of classes are properly chosen, the output codes can tolerate errors in the codeword bits or corresponding classifiers, and hence improve accuracy in an ensemble of classifiers, where the codeword is the decoding decision of the new string within a correction distance, e.g., Hamming distance.

ECOC could improve decision trees and neural networks on several multi-class problems from the UCI repository, but ECOC will not work with classifiers that use local information, i.e., the k-NN classifier. The reason for the failure of ECOC in working with k-NN classifier is that the errors are correlated across the two-class learning problems. Figure 2.3 represents one of a possible coding matrix for a 10 class problem. There are at least three possible ways to produce a coding matrix [35], i.e., deterministic, random, and trained.

2.3.4 Random Subspace Methods

The *Random Subspace Method* (RSM) is the combining technique proposed by Ho [25]. In the RSM, n data sets are created by randomly sampling features from the original training feature space. To be more specific, let each training pattern be described by a p-dimensional vector (or equivalently p features), then we randomly select $r < p$ features from the p-dimensional data set. This way, one can obtain N data sets and, each data set consists of r-dimensional random subspace of p-dimensional feature space. For each data set, we train a base classifier to distinguish all classes. After training n classifiers, we combine their output using simple voting similar to Bagging.

Bagging, boosting, and RSM are designed for, and usually applied to Decision Trees (DT), where they often produce an ensemble of classifiers, which is superior to a single classification rule. However, these techniques may be performed well for classification rules other than DTs. They can be applied to perceptrons, k-NN classification rules. Moreover, they can also be used with LDA and linear classifiers [36].

Recently, the theory of *stochastic discriminant* can be used to describe why RSM and other MCS methods can give large improvements over using a single classifiers. In the SD theory, the degree of enrichment measures how well a model is able to capture a subset of features of class i compared with class j. The model is called *projectable* if its classification representation defined by given training data can be generalized to unseen samples. The definition of uniformity condition is that every data point should be covered by the same number of weak models (for further detail see [37] and references therein).

While Bagging, Boosting, and other SD methods start with highly projectable classifiers with minimum enrichment and seek optimization on uniformity, RSM starts with guaranteed enrichment and uniformity, and seeks optimization on projectability. It should be noted that some MCS methods (e.g., Adaboost, Bagging, and RSM) are related to the theory of

The algorithm:

Let $D_t(i)$ denote the weight of example i in round t.

Initialization: Assign each example $(x_i, y_i) \in E$ the weight $D_1(i) = 1/n$.

- For t = 1 to T:
  - Call the weak learning algorithm with example set E and weight s given by D_t.
  - Get a weak hypothesis $h_t : X \rightarrow Y$.
  - Update the weights of all examples.
- Output the final hypothesis, generated from the hypotheses of rounds 1 to T.

Updating the weights in round t: $D_{t+1}(i) := \frac{D_t(i)exp(-\alpha_i y_i h_t(x_i))}{Z_t}$, where

- Z_t is chosen such, that $D_t + 1$ is a distribution.
- α_t is chosen according to the importance of hypothesis h_t.
  For $h_t : X \rightarrow \{-1, +1\}$ usually α_t is chosen as $\alpha_t := \frac{1}{2} \log \frac{(1-r_t)}{r_t}$,
  where $r_t = \sum_{i=1}^{l} D_t(i) y_i h_t$.

The final hypothesis $H : X \rightarrow \{-1, +1\}$ is chosen as

$$H(X) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)).$$

Figure 2.1 The Adaboost algorithm.

Table 2.3 Error correcting output code: an exhaustive set of codewords.

class	h_0	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}	h_{11}	h_{12}	h_{13}	h_{14}
0	1	1	0	0	0	0	1	0	1	0	0	1	1	0	1
1	0	0	1	1	1	1	0	1	0	1	1	0	0	1	0
2	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
3	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1
4	1	1	1	0	1	0	1	1	0	0	1	0	0	0	1
5	0	1	0	0	1	1	0	1	1	1	0	0	0	0	1
6	1	0	1	1	1	0	0	0	0	1	0	1	0	0	1
7	0	0	0	1	1	1	1	0	1	0	1	1	0	0	1
8	1	1	0	1	0	1	1	0	0	1	0	0	0	1	1
9	0	1	1	1	0	0	0	0	1	0	1	0	0	1	1

stochastic discriminant by nature. Furthermore, some methods give well-enriched weak classifiers (e.g., Adaboost and RSM), while some usually give strong learner (e.g., Bagging and ECOC).

2.4 Recent Formal Definitions of Multiple Classifier Systems

In this chapter, we first briefly describe the classical theory that can be used to describe why the family of MCS classifiers can give large improvements over using a single classifiers. We also address two important equivalencies of multiple classifier systems from the perspectives of Monte Carlo Methods and the functional entropy. The first equivalence is based on the representation of MCS as an integral estimation of the Monte Carlo Methods. The second equivalence is based on the Kolmogorov entropy estimation (optimal encoding). Although the issue of multiple classifier formation is somewhat akin to the stochastic vector quantization [38, 39] and the source-channel model in information theory [8, 9]. However, it is more practicable to exploit the meaning of the latter equivalence for the explanation of its interpretations to information transmission through a noisy channel. These two new formal definitions would provide us some insight information that may be useful for describing the success of this family.

2.4.1 Classical Theory: Bias/Variance Dilemma

In statistical regression estimation, there are two parameters that contribute to the generalization. Bias is the first parameter, which is characterized as a measure of a predictor's ability to generalize correctly to a test set once trained. The second parameter is variance, which can be characterized as a measure of the extent to which the same results would have been obtained if a different set of training data were used.

Recall that in a regression setting, it is possible to decompose the prediction error from using $\widehat{Y}$ to estimate Y, in the following way:

$$E(Y-\widehat{Y})^2 = E\left(Y-E(Y)\right)^2 + \left(E(Y)-E(\widehat{Y})\right)^2 + E\left(\widehat{Y}-E(\widehat{Y})\right)^2. \qquad (2.1)$$

Equation (2.1) is a very useful decomposition. In this decomposition representation, the first term is the variance of Y. The second term is the power of the bias of the estimate $\widehat{Y}$, while the last term is the variance of the estimate. The contribution of this equation is so simple and important, since they guarantee that averaging random variables is always good thing to do (see [p.60, [35]] for details)

Also in this decomposition representation, there are two type of prediction errors: irreducible and reducible errors. The variance of Y is the irreducible error in which it is beyond our control. However, the bias and variance of $\widehat{Y}$ are functions of our estimate and can therefore potentially be reduced.

2.4.2 Relevance to Optimal Encoding of Concepts

Previously, we explored *Bias/Variance dilemma* where the concepts of bias and variance are generalized to classification problems. This approach has the advantage of being relative intuitive and it has produced some interesting results. However, so far, it has failed to address any construction strategies for well-coveraged classifiers or competitively embedded prediction optimization.

Here, we explore a new idea that suitable to bridge the gap between source-channel coding and MCS. First, we detail the work of Donoho [40] and Tapia et al [8] in optimal encoding of concepts. Then, we detail the work of Tapia et al [8] in relating MCS to T-repetition code with threshold decoding. Next, we address the extended idea of optimal encoding of concept to develop the new MCS algorithms. This lays some basic and necessary information needed for the extension of MCS toward the coverage optimization of combining strong learners in Chapters 3 and 4. In particular, the benefit of this framework will lead to better understanding of the new MCS based on the framework of joint source-channel coding and two-stage channel coding methods.

2.4.2.1 Kolmogorov Entropy

Let X be the domain, and let Ξ be a class of functions $(\xi(x) : x \in X)$ on that domain. A class Ξ of functions is said to be totally bounded if for every $\epsilon > 0$, there exists a finite number $\mathcal{N}_\epsilon$ of functions $\xi'_u \in \Xi', 1 \leq u \leq \mathcal{N}_\epsilon$ such that the L_p balls of radius ϵ centered at the ξ'_u's cover Ξ. That is, for every $\xi \in \Xi$, there exists $u \leq \mathcal{N}_\epsilon$ such that

$$\sup_{\xi \in \Xi} \inf_{\xi' \in \Xi'} \|\xi - \xi'\|_p \leq \epsilon. \tag{2.2}$$

The *Kolmogorov ϵ-entropy* [40, 41] of Ξ is then by the following definition

$$H_\varepsilon(\Xi) = \log_2 N_\varepsilon(\Xi). \tag{2.3}$$

Particularly, it defines the least number of bits required to specify any arbitrary member of Ξ to within accuracy ϵ. In essence, Kolmogorov proposed a notion of deterministic encoding for *classes of functions*.

To describe deterministic encoding for a class of concepts [8], let's consider a function of concepts ξ belonging to a target class $\Xi : X \rightarrow \{0, 1\}$, and let Ξ be compact for the norm $\|\cdot\|$. In the Kolmogorov's setting, we can think of the concepts Ξ as a class of functions (i.e., decision boundary), to be compressed as numerical arrays B_l with integer index $l \in \{1, \cdots, \mathcal{N}_\epsilon\}$, thus giving the function of classification as in Figure 2.2, the possible Kolmogorov class concept with a specific ϵ can be represented as in Figure 2.3.

This way, the representation of a class of concepts can then be encoded by a sequence of bits, which abstract from a deterministic encoding E from Ξ into a set $\mathcal{B}$ of bitstreams. That is, the elements $B \in \mathcal{B}$ are sequences of zeros and ones, as illustrated in Table 2.4.

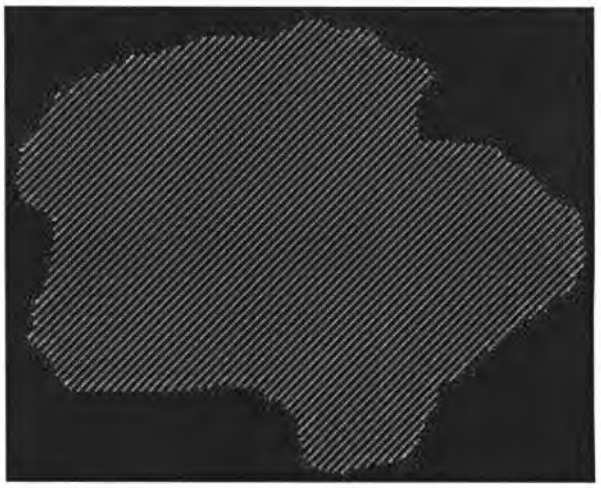

Figure 2.2: Smooth function to be encoded. This could be the classification function of the observed data

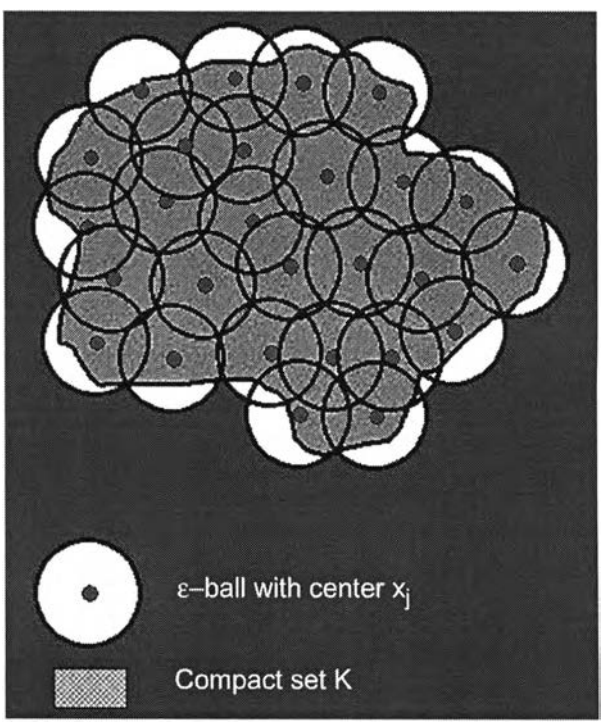

Figure 2.3 Best encoding of the smooth function with distortion ϵ.

Table 2.4 Kolmogorov codebook. Maximum number of bits $= log_2 N_\epsilon(\Xi)$.

Approximate	Code
ξ_1	0000
ξ_2	0001
ξ_3	0010
ξ_4	0011
$\vdots$	$\vdots$
$\xi_{N_\epsilon(\Xi)}$	$b_m \ldots b_2 b_1 b_0$

Thus, an element $E(\xi) \in \mathcal{B}$ is assigned for each concept $\xi \in \Xi$ by E, such that ξ' contains ξ. The reconstruction concept $\widetilde{\xi} \in \Xi$ is then converted from an element $B \in \mathcal{B}$ using a deterministic mapping D.

Generally, $D(E(\xi)) \neq \xi$ and $\|\xi - D(E(\xi))\|_p$ measures the error that occurs in the encoding of concepts. The distortion in such the Kolmogorov ϵ-entropy system is given by

$$d(\Xi, E, D) = \sup_{\xi \in \Xi} \|\xi - D(E(\xi))\|_p. \tag{2.4}$$

On the one hand, the distortion $d(\Xi, E, D)$ means that the deterministic encoding of concepts has distortion $< \epsilon$ if we use at most $\lceil \log_2 N_\varepsilon(\Xi) \rceil$ bits. That is, the larger is the number of bits used to represent the concepts of Ξ (or equivalently the strong learner), the smaller is the distortion achieved. Under the assumption that errors in generalized learning algorithm resembling with the distortion of the encoding of concepts, learning practically becomes a designing problem of constructing the deterministic mapping E_S, especially in terms of approximating the concept function ξ_S and estimating its Kolmogorov entropy $H_\varepsilon^S(\Xi)$. To use too small number of bits, we are subjected to **underfitting**. On the other hand, to use too large number of bits, we are subjected to **overfitting**. We can consider in the latter case that the number of bits transmitted into the abstracted classification channels is larger than the channel capacity.

2.4.2.2 T-Repetition Coding Approach and Its Connection with Adaboost

Based on the framework of encoding of concepts, let a received bitstream B is decoded to be concept a_j by decoding mapping D. In communication term, the mapping $E : C \rightarrow B$ can be modeled as a Discrete Memoryless Source (DMS) with output alphabet X, $|X| = \mathcal{N}_\epsilon$. Let q be a DMS output distribution and let $H(X)$ be entropy characterizing such DMS

$$p_u(a_k) = q_k, \quad k = 1, \ldots, \mathcal{N}_\epsilon, \quad \text{and} \quad a_k \in X \tag{2.5}$$

Let consider the transmission of information symbols from such source through a Discrete Memoryless Channel (DMC) characterized by a finite capacity [42]. As stated in Shannon's Noisy Coding Theorem that reliable transmission of information over a noisy channel can

be achieved by suitable channel coding. The criteria for error rate control in channel coding is that the coded information rate is less than channel capacity.

Recently [8], a linear block code called *T-repetition code* is used for channel coding based MCS scheme. It is very easy to implement T-repetition code in the context of MCS, which we can let a teacher to repeat the target concept T times. Under the assumption of a weak learning algorithm with errors resembling a DMC channel, learning become a decoding problem on received sequence of lossy encoding of target concepts. As discussed in [8], T-repetition code with threshold decoding scheme for MCS is equivalent to Adaboost.

Moreover, let consider a specific case where the expanded concept $\Gamma = F\Xi$ be the transformed concept caused from using a particular linear transform. In general, Γ has a linear dependence between its components and the original concept sequence Ξ. As discussed in [8], $E : \Xi \rightarrow B$ can be interpreted as a quantization process with output alphabet Y, $|Y| = \mathcal{N}_\epsilon(\Xi)$. However, the quantization process in source encoding [43] always makes the components of $\widehat{\Gamma} = E(\Gamma)$ linearly independent. In other words, each components of $\widehat{\Gamma}$ −even in excess of k −gives distinct information on the value of Ξ, where k is the least number of bits required to encode $\mathcal{N}_\epsilon(\Xi)$. This is in fact one of the supporting ideas for the need of joint source-channel coding in the framework of multiple description coding [43, 44] for MCS, which will be presented next in Chapters 3.

2.4.3 Relevance to Variance Reduction Techniques in Monte Carlo Methods

Previously, we describe one of the new formal definitions for MCS. Here, we explore another idea related to the approximation theory for classification function representation of the observed data. First, we detail the work of Friedman and Popescu [28] in casting MCS into Monte Carlo methods. The central idea of supervised learning problems in data mining, machine learning, and pattern recognition is crucially related to function approximation of many arguments.

It is in this framework that *important sampling* strategies become the main ingredient of many popular ensemble methods. Most of the family of ensemble classifiers can be cast within the *important sampled learning ensemble* (ISLE) framework. For example, various randomized methods, e.g., Bagging, random forest, and Bayesian averaging are seen to correspond to random Monte Carlo method based on independent identical distribution (iid) perturbation sampling and MCMC strategies, respectively. In fact, the ensemble of base classifiers produced by these techniques are obtain in a parallel manner; each individual classifier is trained without any information about the others. Non random boosting methods, on the other hand, are seen to correspond to deterministic quasi-Monte Carlo integration techniques. In these techniques, information about the other members of the ensemble can be used for generating the base classifiers.

In general, there are a variety of variance reduction techniques [45] in Monte Carlo methods, which we can group them into several categories based around four main ideas: (1) analytical integration by finding a function that can be integrate analytically and is similar to

the integrand; (2) uniform sampling placement where each subdivision region is uniformly sampled by its corresponding number of sampling; (3) adaptive sampling for controlling the sample density in order to place more samples where the integrand is large or changes rapidly; and (4) correlated estimators using combining samples from two or more estimators whose values are correlated. In detail, based on the above categories, these techniques comprise important sampling and control variates, stratified sampling and Quasi-Monte Carlo, conditional Monte Carlo, common and antithetic variate methods, respectively.

Since the main focus of this dissertation is on the construction strategies for combining strong learners toward the parallel approach, we are thus more interested in reducing variance rather than bias. As a consequence, we next detail an alternative method of utilizing correlated estimators toward common and antithetic variance reduction method. This lays some basic and necessary information needed for the extension of ISLE toward the prediction optimization of combining strong learners in Chapter 5.

2.4.3.1 Importance Sampling

Recently, Friedman and Popescu [28] proposed an alternative view of ensemble learning based on Monte Carlo integration of basic linear model. Consider the problem of ensemble learning, the optimal target concept can be formulated as

$$F(\mathbf{x}) = \int_{\mathbf{P}} \mathbf{a}(\mathbf{p})\mathbf{f}(\mathbf{x};\mathbf{p})\mathbf{dp}, \tag{2.6}$$

where $f(\mathbf{x};\mathbf{p})$ is a base learner (basis function), $\mathbf{p} \in \mathbf{P}$ indexes particular function of x from $f(\mathbf{x};\mathbf{p})$, $a(\mathbf{p})$ is the coefficient of $f(\mathbf{x};\mathbf{p})$, and $F(\mathbf{x})$ represents the best target concept of classification output $\mathbf{y}$ given training samples $\mathbf{x}$ under loss function, L. In statistics, $f(\mathbf{x};\mathbf{p})$ can be any learning functions, i.e., tangent functions (neural networks), multivariate splines (MARS), or decision trees (random forest).

Introducing $I(\mathbf{p}) = \mathbf{a}(\mathbf{p})\mathbf{f}(\mathbf{x};\mathbf{p})$, the integral in (2.6) can also be written as

$$F(\mathbf{x}) = \int_{\mathbf{P}} \mathbf{I}(\mathbf{p})\mathbf{dp}. \tag{2.7}$$

Consequently, one can approximate the integral $F(\mathbf{x})$ with tractable sums $F_M(\mathbf{x})$. In other words, the estimate $F_M(\mathbf{x})$ is unbiased and converge almost surely to $F(\mathbf{x})$ by the strong law of large numbers (very large r). Hence, we obtain

$$F(\mathbf{x}) \simeq \sum_{m=1}^{M} w_m I(\mathbf{p_m}), \tag{2.8}$$

$$\simeq \sum_{m=1}^{M} w_m a(\mathbf{p_m})\mathbf{f}(\mathbf{x};\mathbf{p_m}), \tag{2.9}$$

$$\simeq \sum_{m=1}^{M} c_m f(\mathbf{p_m}). \tag{2.10}$$

There are a variety of techniques for Monte Carlo evaluation of the approximated integral in (2.10), one method is called *perturbation sampling*. To perturbation sampling, sampling locations $\mathbf{p_m}$ should be controlled by a monotonic decreasing function of a predetermined important measure $J(\mathbf{P})$. In other words, the probability density function (pdf) of the sampling location $\mathbf{p_m}$ is defined by the function $g(J(\mathbf{P}))$. For example, the sampling density function $g(J(\mathbf{P}))$ might be as simple as *constant*. When $g(J(\mathbf{P}))$ is constant, it means that a pool of estimators or classifiers is built without accounting for good/important candidates. However, if $g(J(\mathbf{P}))$ is a single *delta function*, then only the best single base learner is accounted. In between, we can balance such two situations by arranging $g(J(\mathbf{P}))$ to take some good candidates into account. As described in Reference [28], the available ensemble learning methods can be effectively explained by the construction strategies of the $g(J(\mathbf{P}))$ and the coefficients c_m. Furthermore, there is a suggestion that the coefficients c_m should be found by solving the set of linear regression equations of $\mathbf{y}$ on M populations of $f(\mathbf{x}; \mathbf{p_m})$ averaging over $\mathbf{x}$.

By considering the equivalence between MCS and the Monte Carlo Methods, we can see how such the efficiency of MCS arise naturally in the framework of the Monte Carlo Methods. As discussed in Reference [28], one can view that most popular methods in ensemble learning are either implicitly or explicitly related to variance reduction techniques in the Monte Carlo methods, especially important sampling. Understanding variance reduction techniques for MCS is thus useful for reducing the degree of subtleness in ensemble learning error.

2.4.3.2 Antithetic and Common Variates

Essentially, the third term of (2.10 is the variance of the estimators. This term can be further reduced if any two estimators in the ensemble have appropriated correlations. Here is the idea we explore for these appropriated correlations. We first consider the case of simple averaging, then generalize it to weighted combining. In particular, this part is somewhat akin to variance reduction methods in Monte Carlo methods.

The idea of antithetic (common) variates is to find two functions $g(X)$ and $h(Y)$ whose values are negatively (positively) correlated, and add (subtract) them, respectively. Consider a simple case where P_X and P_Y are the known cumulative distribution functions (cdf) of two random variables X and Y. The average of the two random variables will be the random variable with variance

$$Var\left[\frac{1}{2}(X+Y)\right] = \frac{1}{4}Var(X) + \frac{1}{4}Var(Y) + \frac{1}{2}Cov(X,Y). \tag{2.11}$$

We can see that the variance of $\frac{1}{2}(X+Y)$ is minimized when the covariance $Cov(X,Y)$ is strongly negative. Assume that both X and Y are generated by the inverse transform method

$$X = P_X^{-1}(U_1), \tag{2.12}$$

and

$$Y = P_y^{-1}(U_2),$$

where U_1 and U_2 are uniformly distributed on $[0,1]$. The method of antithetic variates is used when $U_2 = 1 - U_1$. It is known that when antithetic variates are used, then $Cov(X,Y)$ is strongly negative and $Var\left[\frac{1}{2}(X+Y)\right]$ is minimized.

Consider now the average of a pair of two random variables (i.e., estimation or classification functions)

$$E\left[\frac{1}{2}\left(g(X)+h(Y)\right)\right],$$

where both random variables are real-valued measurable functions of the two random variables X and Y, respectively. Furthermore, assume that the marginal cdf P_X and P_Y are known. Let's define

$$g^*(U_1) = g\left[P_X^{-1}(U_1)\right], \tag{2.13}$$

and

$$h^*(U_2) = h\left[P_y^{-1}(U_2)\right].$$

As proved in the Reference [46], if g and h are monotonic in the same direction then

$$\begin{aligned} Var\left[\frac{1}{2}\left(g(X)+h(Y)\right)\right] &= Var\left[\frac{1}{2}\left(g(P_X^{-1}(U_1))+h(P_Y^{-1}(U_2))\right)\right] \\ &= Var\left[\frac{1}{2}\left(g^*(U_1)+h^*(U_2)\right)\right] \end{aligned} \tag{2.14}$$

is optimal by the use of the antithetic variates.

Note that the above statements are always true for both the univariate and multivariate cases. In multivariate case, giving $P_X(X)$ and $P_Y(Y)$ by d-dimensional distributions

$$P_X(X) = \prod_{i=1}^{d} P_{x_i}(x_i) \quad and \quad P_Y(Y) = \prod_{i=1}^{d} P_{y_i}(y_i,) \tag{2.15}$$

where $X = P_X^{-1}(U_1) = \left\{P_{X_1}^{-1}(U_1^1), ..., P_{X_n}^{-1}(U_1^d)\right\}$, and $Y = P_Y^{-1}(U_2)$ $= \left\{P_{Y_1}^{-1}(U_2^1), \ldots, P_{Y_n}^{-1}(U_2^d)\right\}$, then $Var\left[\frac{1}{2}\left(g(X)+h(Y)\right)\right]$ is minimized, if g and h are monotonic in the same direction.

Since there is not any very obvious way in which to apply the antithetic variates in the more general multidimensional case, we can transform the antithetic univariates to a unit cube (of the appropriate dimensionality). In particular, in the three-dimensional unit cube the analog of the one-dimensional function:

$$\frac{1}{2}[g_1(u) + g_2(1-u)]$$

contains 8 terms or 2^3 terms:

$$\begin{aligned} &\frac{1}{8}[g_1(u,v,w) + g_2(1-u,v,w) + g_3(u,1-v,w) \\ &+g_4(u,v,1-w) + g_5(1-u,1-v,w) + g_6(1-u,v,1-w) \\ &+g_7(u,1-v,1-w) + g_8(1-u,1-v,1-w)], \end{aligned} \tag{2.16}$$

Table 2.5: Relationship between common-antithetic variates and monotonicities of function g and h.

monotonicities of g and h	Difference of RVs	Sum of RVs
same direction	Common	Antithetic
opposite direction	Antithetic	Common

where all of g_i are monotonic in the same direction, and u, v, w are 1-dimensional random variables.

So far we have showed that if g and h are monotonic in the same direction, the use of antithetic variates for the average of two real-valued measurable functions, that is $U_2 = 1 - U_1$, can effectively reduce the variance. It is straightforward to show that if g and h are monotonic in the opposite direction, the use of antithetic variates for the mean difference of two real-valued measurable functions, that is $U_2 = 1 - U_1$, can effectively reduce the variance. This can be illustrated by replacing $h(Y)$ by $h^{'}(Y) = -h(Y)$ and subtraction $h^{'}(Y)$ to $g(X)$ as in (2.14). Hence, we obtain that g and $h^{'}$ are monotonic in the opposite direction.

Having established that antithetic variates are the effective methods for reducing the variance of either the average or the mean difference of two real-valued measurable functions $\frac{1}{2}[g(X) - h(Y)]$. Depending on the monotonicity directions of the functions g and h, we can conclude that the variance of the average of two functions will be optimal, if g and h are monotonic in the opposite direction. Thus, the use of common variates, that is $U_2 = U_1$, can effectively reduce the variance. In the same way for the case of the mean difference of two functions, one can prove that if g and h are monotonic in the same direction. Similarly, the use of common variates can effectively reduce the variance (see the proofs in Reference [47]). Here, we can give a summary of conditions for the uses of common and antithetic variates in Table 2.5.

Further extension of the preceding cases is given in Reference [46]. He showed that the mixed common and antithetic variates, called the vector of antithetic-common variates (VACVs), can be also constructed for multidimensional cases. For example, let $g(u_1, v_1)$ be monotonic increasing in u_1 and monotonic decreasing in v_1 and $h(u_2, v_2)$ be monotonic increasing in both u_2 and v_2. Then the vector $u_2 = (u_2, v_2) = (u_1, 1 - v_1)$ is the optimal VACVs for the use of common variates. Naturally, we can generalize (2.16) for the use of antithetic variates in multidimensional problems to VACVs, as shown in the following

$$\Omega = \frac{1}{M} \sum_{i=1}^{M} g_i \left(U_i \right), \tag{2.17}$$

where $M \leq d$, and an appropriate dependence between U_i are induced and g_i are given. Then, the variance of (2.17) is less than the variance in the case when the $U_1, ..., U_M$ are independent. At this point in the discussion, we can see the equivalence between the use of

VACVs and simple average combination rule in ensemble learning.

In practical, one can admits correlated variates to have less optimality than the VACVs in a given set of learning examples. Because of the link that clearly exists between antithetic-common variates and variance reduction techniques, it is reasonable to define the *partial antithetic-common variates* (PACVs) as a variance reduction techniques if it can be determined that both the monotonicity assumption and the dependence condition between correlated variates are in some sense "satisfied". As a consequence, one might expect that the averaging term in (2.17) can then be replaced by optimal weight combining rules.

It can also be interpreted that variance reduction method based on the antithetic and common variates is somewhat akin to the least square estimates of the combining-weights. In Chapter 4, we review some least square methods that mainly focus on the variance reduction of combining multiple classifiers. This is corresponded to the combining strong learner approach.

2.5 Rationale for a New Approach

As shown in our review of methods for MCS, many alternative solutions and interpretations may exist for a particular recognition problem. It should be noted that beside the above interpretations, the necessary of ensemble methods is to overcome three key shortcomings of standard learning algorithms, i.e., the statistical, computational, and representational issues [14]. This is similar to our more elaborated discussion on how to achieve good statistical or representational models of the observed data, in which there is too little dentinal given in [14]. We are now summarizing the advantages of using new extensions of MCS as follows:

Signal-domain Channel Coding Approach. Leading to the motivation on the use of source-channel modeling of MCS (or equivalently transmission of optimal encoding concepts through a noisy channel), one can easily relate any successful channel coding methods, e.g., joint source-channel coding, to further improve the classification accuracy of the MCS. Joint source-channel coding implemented here is related to *multiple description coding models*, which is also considered as signal-domain channel coding approach that is very applicable for both the erasure and noisy channel. This approach is somewhat akin to *multiple descriptors for a mixture of features*. In contrast to the conventional idea, we use multiple descriptors to train an ensemble of classifiers. Each classifier response to only partial information that characterized a pattern. When a linear combination of all classifier outputs is made, then the representation of concept can be finely approximated. The proposed method here is more deterministic than the most of the conventional MCS methods [13, 25, 36, 48]. Moreover, the proposed method is systematically related to wavelet based multiple description coding model that it can takes several advantages of the wavelet transforms for pattern recognition.

Generalized Code Concatenation Approach. Generalized code concatenation approaches is a very efficient code scheme widely known in coding theory. It is also known

that generalized concatenated code has several advantages over the classical concatenation method [49, page 288]. Classical code concatenation approaches is somewhat akin to *hybrid methods of ISLE*. Thus, a code concatenation-like output code can further improve the classification accuracy over a single output code of the same length.

Prediction Optimization Scheme. To motivate the use of variance reduction techniques in ensemble learning analysis, one should attempt to see how well they can be used to clarify learning methods and their optimal choices for specific problems, especially in some difficult cases. One of the very difficult problems in pattern classification is related to the problem of prediction (decision) dependence, which is of important and difficult, remains controversy, and has received very little attention in recent literature. This is also one of the major concerns in variance reduction techniques in Monte Carlo methods.

Having believed that variance reduction techniques play an important role on the learning and test efficiency of MCS, we however, empirically found that there are no single variance reduction technique that is generalized well for all classification problems. As widely accepted, Adaboost always performs much better than any other MCS, if we do not imposed any constraints on the complexity of learning and testing of MCS. In other words, if we do not concern with the efficiency of the variance reduction techniques, important sampling-based multiple classifier systems may be the most-favorable choice in the MCS list. Except for the reasons that important sampling is an efficient technique for the Monte Carlo Methods and Adaboost is related to important sampling in term of sequential sampling in the Monte Carlo methods [28], the use of Adaboost for multiclass problems [50] is reported to be very computational expensive, especially when using with less powerful *weak* learners. Furthermore, it is more susceptible to noise and quickly overfit a data set [27]. One of the explanations owed to the trade–off between classifier's accuracy and its simplicity. In fact, one cannot in general tell what is the feasible classifier set in the accuracy–complexity space [27, 51]. Classifier strength, size of the classifiers, and nature of the algorithms' outputs [27, 52] were mentioned as three factors that control the overall accuracy of classification design. Besides the three factors, we believe that data regularity is also a key factor on the overall accuracy of designed classifiers.

Thus, we should explicitly design an algorithm by keeping in mind the important of data regularity in conjunction with classifier strength, size of classifier, and the nature of the algorithms' output. In other words, we should use an algorithm that can manipulate data in such a way that we can avoid not to choose too small classifier that is subject to underfitting, or too large that is subject to overfitting. Using different approach from Adaboost, next, an alternative algorithms using classifiers with *strong* classification power rather than weak will be presented. Since each strong classifier is trained with respect to a particular data regularity (a proper coordinate system) in a diverse fashion and each learning concept is large enough to cover the target concept, the convergence rate of our proposed algorithms become faster than Adaboost. In fact, this dissertation presents new experimental evidences against the utility of ensemble of weak classifiers.

In Chapter 4, we review some least square methods that mainly focus on the variance reduction of combining multiple classifiers. This is corresponded to the combining strong learner approach (as opposed to combining weak learner approach, for example, boosting, where bias and variance reduction are both considered).

Bayesian and Incremental Learning Framework. It is widely known that neural networks are universal approximator. One of the possible solution in parallel coverage construction of MCS is thus inspired by exploiting the interpolation power of the neural networks. In particular, multiresolution descriptors (or equivalently subbands) are used to in a multiple classifier system to highlight different information needed to characterize a pattern. This is another type of MCS that can be cast into Bayesian averaging framework. Moreover, it can be easier related this method to the incremental learning framework in the sense that a new output unit is created whenever the new subband information is provided.