ABSTRACT

Title of Document:	OPTIMIZED FUSION TECHNIQUE FOR FAULT DIAGNOSTICS.
	Surya Kunche, Master of Science, 2013
Directed By:	Prof. Michael G. Pecht, Department of Mechanical Engineering

Classification algorithms have been widely used to solve data-driven fault diagnostics problems. The number of classification algorithms used has been increasing in recent years. Each classification algorithm has its own strengths and weaknesses, and the accuracy of classifiers changes with the different features used for training. As a result, traditional methods of selecting an appropriate classification algorithm, including domain expertise and trial and error, are becoming complex and difficult to employ. Classifier fusion has been used to solve this problem of selecting an appropriate diagnostic algorithm, and it also improves the generalizability of an algorithm. The performance of a classifier fusion algorithm is governed by the combination rule adopted for fusing multiple classifiers and how the bias and variance are balanced by the combination rule. However, research still needs to determine which combination rule optimally balances the bias and variance during classifier fusion. Therefore, this research develops a fusion methodology that combines the classifiers by balancing the bias and variance. This methodology reduces the number of false negatives and positives, thereby improving the overall accuracy of the algorithm for fault detection. A cost function that considers bias and variance errors was developed to evaluate the performance of the algorithm. Sequential quadratic programming–based optimization was employed to find the optimal combination of classifiers and balance of bias and variance. The developed algorithm was used for fault diagnosis of analog circuits, and the results indicate that the developed fusion approach improved diagnostic accuracy over existing classifier fusion techniques.

OPTIMIZED FUSION TECHNIQUE FOR DIAGNOSTICS

By

Surya Tej Kunche.

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2013

Dedication

I dedicate this work to my parents, friends and family.

Acknowledgements

I am grateful for the support of my advisor Professor Michael Pecht. He graciously provided academic advising, and financial support that allowed me to undertake this work. I would also like to thank my committee members; Professor Prof. Patrick McCluskey and Professor Peter Sandborn. I would also like to thank Dr. Chaochao Chen for constantly guiding me during this work. I would also like to thank Dr. Diganta Das and Dr. Michael Azarian for the comments during the morning meeting.

I would also like to thank the more than 100 companies and organizations that support research activities at the Center for Advanced Life Cycle Engineering (CALCE) at the University of Maryland annually and, specifically, the CALCE prognostics and Health Management Consortium members.

I would like to thank my lab mates who were helpful in critiquing this work during the morning meeting presentations. This has been really helpful in continuously improving this work.

I am deeply indebted to my parent for their love and support and to my sister and brother in law who has always made me feel at home in Maryland.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

This chapter presents the relevant background information and motivates the problem addressed in this thesis. The opportunities for research are presented and an overview of this thesis is presented.

Fault diagnosis is the process of identifying faulty behavior in systems. Classification algorithms have been widely used to solve data-driven fault diagnostics problems. Classification is a process of identifying to which category a test observation belongs to based on the historically observed features whose category is known. When the category of the training data is known supervised learning is typically used to train a classifiers. The task of supervised learning is to constructor learn the underlying relationship between the feature attribute and the category attribute (healthy or faulty) based on the available training data to predict health state of the system.

1.1 Motivation

Prognostics and health management (PHM) is an enabling discipline consisting of technologies and methods to assess the reliability of a product in its actual life cycle conditions to determine the advent of failure and mitigate system risk [1]. Diagnostics plays an important role in PHM regimen by identifying faulty behavior and isolating these faults.

Classifiers are used to identify faults in data driven diagnostics. Any classification problem is an ill posed problem, which implies there are multiple possible hypothesis for a given set of input feature. While a number of classification algorithms have been developed, each of the developed algorithms has their advantages and disadvantages. Their performance or classification accuracy varies with the kind of features chosen for training, amount of training data available and the model characteristics of the classifier used. The performance of a classifier is also referred to as the generalizability of a classifier. Generalizability implies good prediction ability on observed features that the classification algorithm has not been trained on [2]. As the number of diagnostic algorithms (classification algorithms) has been increasing, the classification algorithm selection process has become complex and difficult to employ. Classification is an ill posed problem, which implies there could be multiple solutions to the same training data set. Hence to solve this problem, classifiers have an innate property known as the inductive bias. Inductive bias of the classifier is defined as the set of assumptions that a classifier makes in order to solve the classification problem. For example a support vector machine (SVM) solves a classification problem by finding the maximum margin separating hyperplane [3]. This hyperplane is solves as a structural risk minimization problem as proposed by Vapnik [4]. This structural risk minimization formulated as shown in equation 1. Equation 1 is solved to find the optimal hyperplane $w^T x_i + b$ where w is the normal to the hyperplane and $|b|/\|w\|$ is the distance of the hyperplane from the origin of the coordinate system [4].

$$\min J(w, \varepsilon) = \frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \varepsilon_i \tag{1}$$

$$s.t. \qquad y_i(w^T \emptyset(x_i) + b) \geq 1 - \varepsilon_i \; when \; y_i \epsilon \{-1, 1\} \; and \; \varepsilon_i \geq 0 \tag{2}$$

where ε_i is the slack variable or the distance margin introduced to allow misclassifications, and γ is the penalty or the cost for the misclassifications. The function $\emptyset(x_i)$ is a mapping of x_i to a higher dimensional space. The maximum separable hyperplane so obtained separated the classes in the two class classification problem as shown in Figure 1. Inductive bias is not a perfect learning theory and leads to errors when solving the learning problem. The errors thereby caused are called the inductive errors of the classifier. For the support vector machine the errors so caused are as illustrated in Figure 2

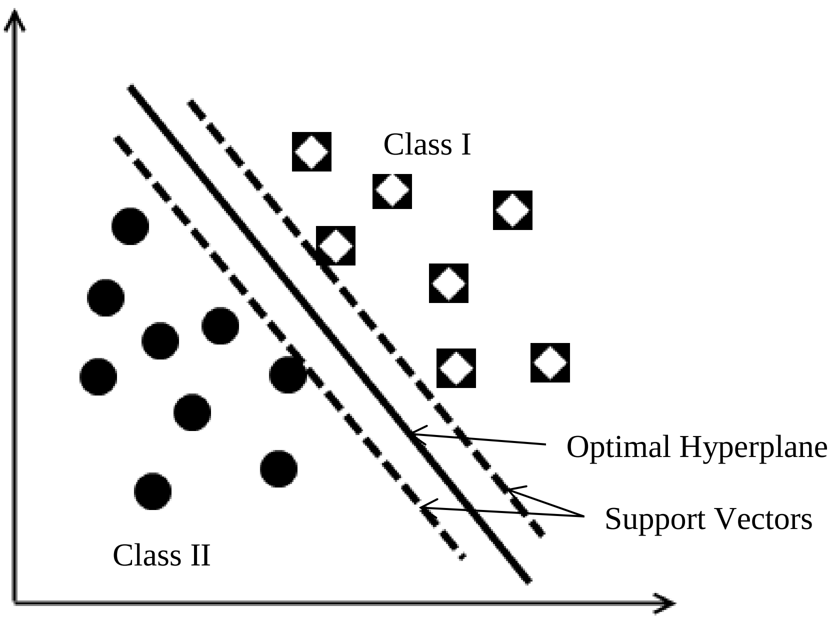

Figure 1 SVM Classification

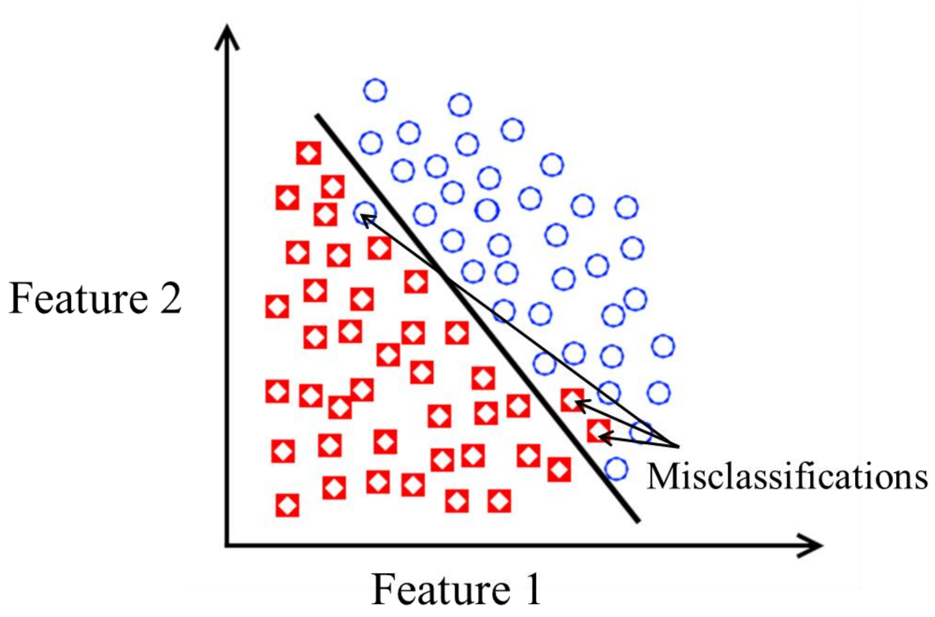

Figure 2 Inductive Bias Error in SVM

1.2 *<u>Fusion based Learning</u>*

Classifier fusion has been widely employed to overcome the inductive bias problem in single classifier based system. In classifier fusion an ensemble of base classifiers are trained and the prediction of each classifier in the ensemble are combined to build the predictive model [5]. A typical framework for a classifier fusion technique is as shown in Figure 3.

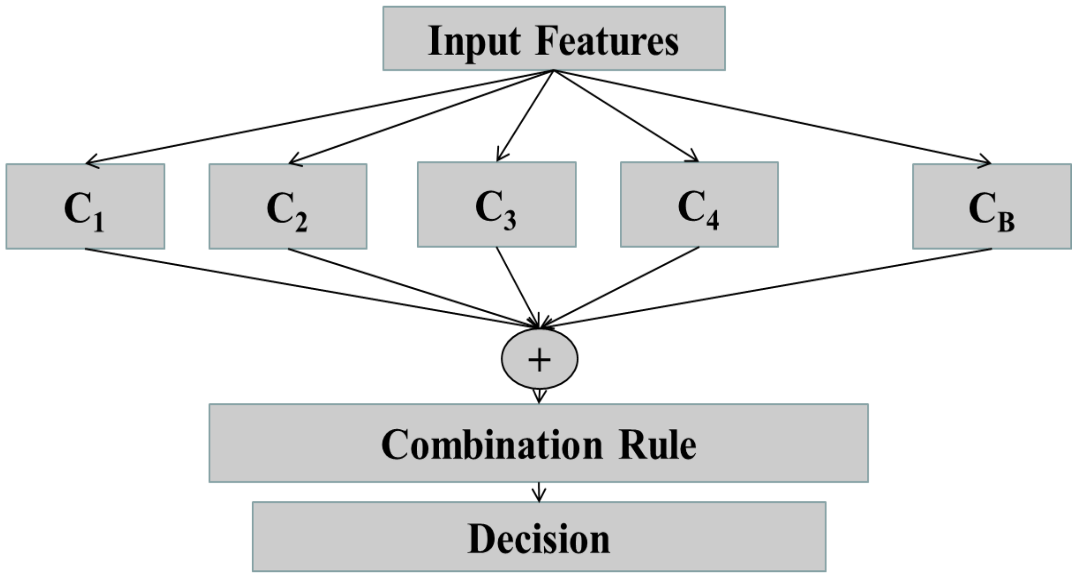

Figure 3 Classifier Fusion Framework

The typical classifier fusion strategy has two stages the diverse ensemble generation stage and the ensemble fusion stage. These two stages are discussed in detail in the following two sections.

1.2.1 Diverse Ensemble Generation

An ensemble of classifiers are said to be diverse if each classifier makes errors on different observed feature instances of training data [6]. Classifier fusion theory is based on the idea of training an ensemble of classifiers which do not make the same prediction mistakes as others and when they are combined these complementary features can be used to overcome inductive errors of a classifier. Hence, diversity plays a very important role in classifier training. The errors that classifiers are said to be complementary.

Brown *et al.* [8] provided a mathematical account of the role of diversity in ensemble learning and how it helps to improve classification accuracy. The authors concluded that the prediction accuracy of an ensemble based classifier is dependent on the diversity of the classifiers. Theoretically, the more diverse the classifiers are, the less correlated the classifier outputs are with each other. As a result, the prediction of each classifier of the ensemble could be complementary to each other, which implies when one classifier makes a prediction error, other classifiers could be correct or vice versa. The complementary nature of these classifiers helps in offsetting potential errors, thereby providing greater generalizability for these algorithms.

Diversity can be achieved by two different means when classifiers are trained. It can be achieve by training the classifier with different training data. Techniques that fall under this category are bootstrapping, bagging and boosting [7]. Diversity can also be achieved changing the classification algorithm used for training [7].

1.2.2 Ensemble Fusion

Once a diverse set of classifiers are trained, the goal of ensemble fusion is to take advantage of the complementary feature of the trained classifiers in the ensemble to overcome their inductive errors. This is essentially done by using a combination rule as shown in Figure 3. A detailed review of the classifier fusion techniques is done in the literature review section.

1.3 Bias-Variance Dilemma

As discussed in previous section the performance of an ensemble of classifiers is dependent on how each classifier is trained in the ensemble and how they are combined using the combination rule. But, Geman *et al.* [9] were the first to study the underlying factors affecting the performance of classifier fusion based learning. The authors found that the generalizability of an ensemble of classifier is governed by two errors the bias error and the variance error. The bias error is defined as the difference between the predicted class and actual class of a particular observed feature [10]. On the other hand variance of a classifier is a measure of the variance of prediction of the classifier with respect to the expected prediction [10]. Given a training set $T_k\,\{(x_i, y_i), i = 1,2,\cdots,N\}$ for classifier k, where $k = 1,2,\cdots K$, x_i is the feature vector and y_i is the corresponding class vector. Let $\widehat{y_k}$ be the classifier trained on training set T_k and $\widehat{y_{k\iota}}$ is the prediction of classifier k for input feature x_i.The bias and variance of a classifier are as shown in Equation 3 and Equation 4 respectively.

$$Bias\ for\ classifier\ k = \sum_{i=1}^{N} |y_i - \widehat{y_{k\iota}}| \tag{3}$$

$$Variance = \sum_{i=1}^{N} (\widehat{y_{k\iota}} - E(\widehat{y_\iota}))^2 \tag{4}$$

$$E(\widehat{y_\iota}) = \frac{1}{K} \sum_{k=1}^{K} \widehat{y_{k\iota}}\,; i: 1 \to N \tag{5}$$

$$total\ error = \ bias^2 + variance \tag{6}$$

Geman et al. [9] have mathematically proved that the total error of an ensemble of classifiers is as shown in Equation 6. Both the error quantities changes as the model complexity of a classifier changes as shown in Figure 4. Model complexity of a classifier depends on how the training parameters of a classifier are chosen. For example in a least square estimation the model complexity depends on the order of polynomial chosen of least square estimation. In a neural network the complexity increases with the increase in number of hidden neurons in the neural network.

Geman *et al.* [9] have also found that the bias and variance errors are inversely proportional as shown in Figure 4. Hence as one component increases the other component decreases. An optimal balance between the bias and variance is leads to reduction in total error and hence good generalizability as shown in Figure 4.

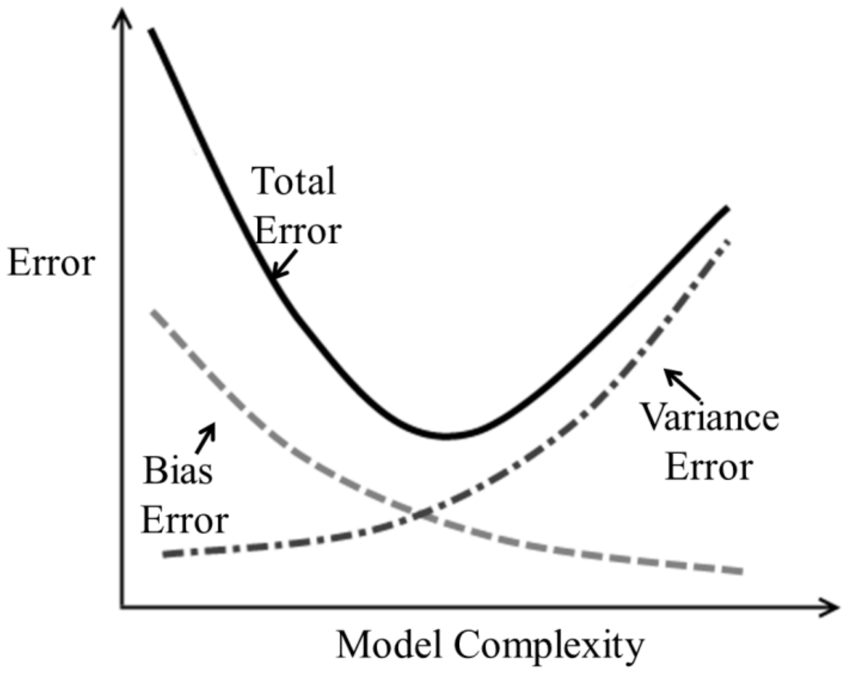

Figure 4 Error change as a function of model complexity

1.4 Literature Review

Diversity in an ensemble of classifiers can be achieved either by randomization methods or by metric based methods. Randomization-based methods are the most widely used techniques for diversity-generation, where diversity is achieved by varying the training data i.e. supplying each classifier with a different set of manipulated training data (for example, training a classifier only with part of the training data). When a classifier is trained with a manipulated training data set, it typically generates prediction results that are diverse. Commonly used methods for manipulating training data are bagging [11] (resampling training data using bootstrapping and building a classifier on these resampled training sets), and boosting

[12] (training an ensemble of classifiers on subsamples of training data that other classifiers in the ensemble misclassified). Diversity is also achieved by changing the model parameters of a classifiers being trained; for example, in neural networks diversity can be achieved by changing the initial weights, the number of hidden neurons, the activation function and the training algorithm [8] and for a support vector machine this is achieved by choosing the kernel function. Some researchers have proposed the use of an evolutionary algorithm to achieve the optimal amount of diversity during the training phase [13][14][15].

Once diversity is achieved for the trained classifiers, a combination rule is used to combine the classification results. The most common means of classifier fusion include averaging [16][17], majority voting [18][19][20], weighted majority voting [21], and a localized fusion [22][23] based approach that improves the weighted majority voting algorithm by evaluating the performance of classifiers in the neighborhood of the test points. Bonissone *et al.* [24] proposed a fusion methodology based on Cartesian and regression trees that reduce the computation time in the localized fusion approach.

<u>*1.5 Research Gaps*</u>

In averaging based fusion technique [16][17] prediction of the classifiers are combined by averaging all the predictions. Given an ensemble of classifiers $\widehat{y_1}, \widehat{y_2}, \cdots, \widehat{y_B}$, where B is the number of classifiers in the ensemble the averaging based fusion rule combines the classifiers based on the equation as shown in equation (7). The drawback of this approach is that the fusion technique does not consider the performance of each classifier in the ensemble. This implies classifiers which have poor performance are weighted equally as classifiers which have good performance, hence making this fusion techniques much weaker than single classifier based systems.

$$\hat{y} = \frac{1}{B}\sum_{b=1}^{B}\widehat{y_B} \tag{7}$$

The majority voting based fusion technique polls the multiple classifiers in the ensemble. The highest polled classifier is then chosen as the class of the observed feature. This approach has the same drawback as the averaging based fusion technique.

Weighted majority voting [21], localized fusion [22][23] based approach and the Cartesian and regression tree based approach proposed by Bonissone *et al.* [24] utilized the bias compensation technique where in the classifiers are weighted by inverse of the bias error of a classifier as shown is equation (8). Where e_i is the bias of the classifier.

$$\hat{y} = w_1\hat{y}_1 + w_2\hat{y}_2 + .. + w_B\hat{y}_B \tag{8}$$

$$w_i = \frac{1}{e_i} \tag{9}$$

The difference between localized fusion technique and weighted fusion technique is that, the localized fusion technique computed the bias only in the neighborhood of the feature whose class needs to be predicted by the classifier. This is illustrated as shown in Figure 5. The neighborhood is identified by finding the nearest neighbors. Figure 5 illustrates the 3 nearest neighbors.

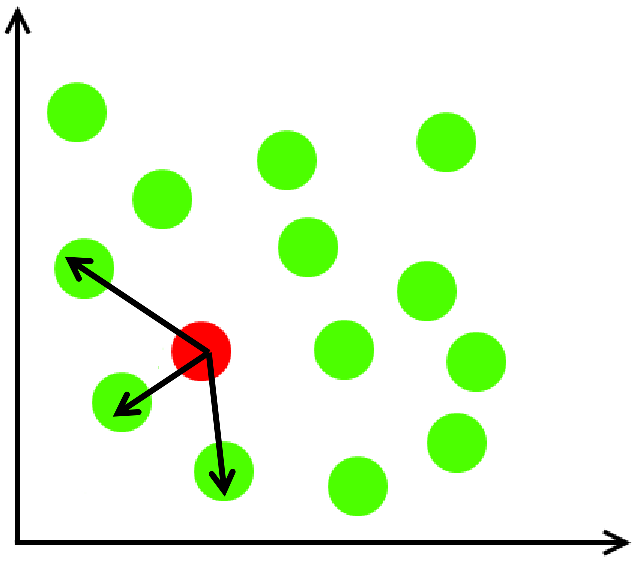

Figure 5 Bias Computation in Localized Fusion

Chapter 2: Developed Fusion Algorithm

In this chapter the developed fusion technique is discussed in details the three stages of the classifier fusion problem (i) training (ii) fusion parameter computation and (iii) classifier fusion are discussed in details. The sequential quadratic programming technique used to optimize the formulated objective function is discussed in detail in chapter 3.

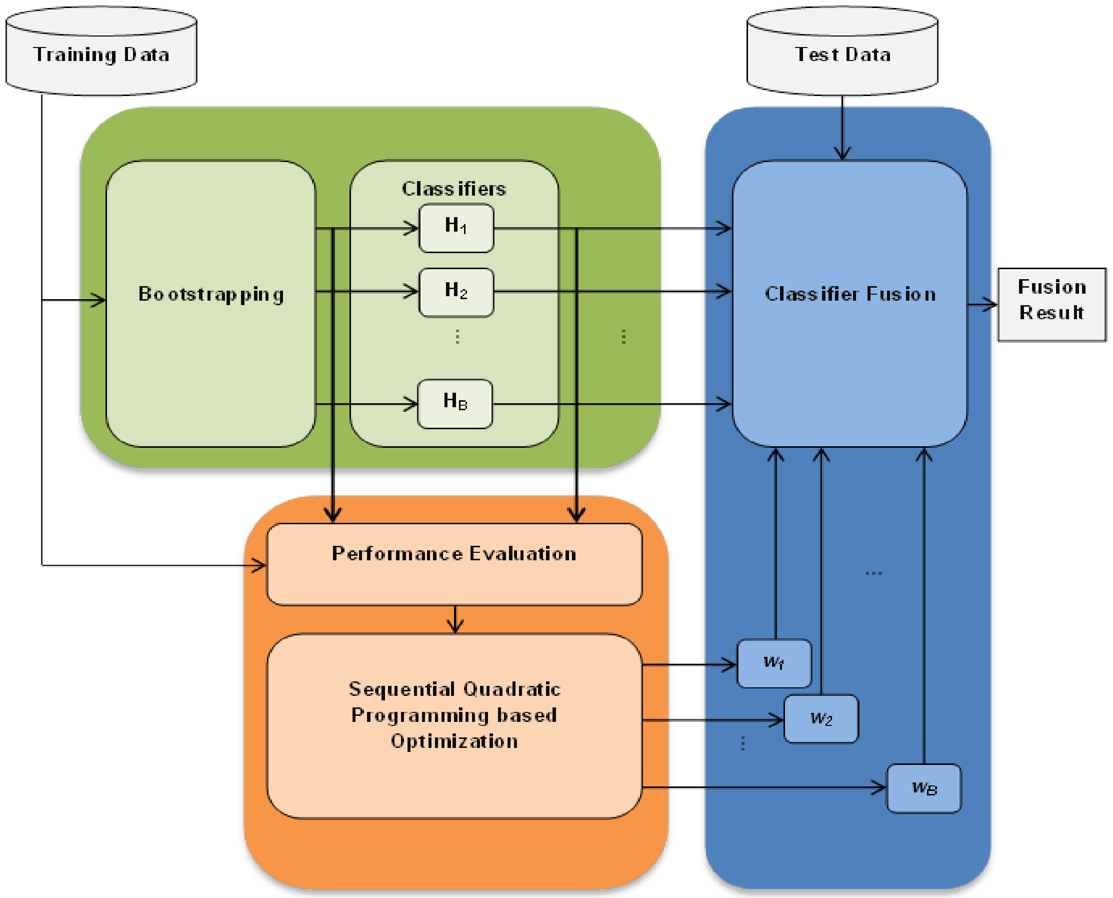

Figure 6 Developed Fusion Framework

2.1 Algorithm Training

Diversity in this ensemble learning framework has been achieved by two means. The first method used for generating diversity is by using bootstrapping and the second method is by using different classification algorithms. The following subsections describe the steps involved in achieving these two means of diversity.

2.1.1 Bootstrapping

The bootstrapping method was originally proposed by Efron [25]. When bootstrapping is used, resampling of training data is done such that sample observations are picked randomly by replacement from the original training data to form new training data sets which are the same size as the original training data set [26]. As shown in Figure 4, considering an original training data which has 5 observations, the bootstrapped training sets also consist of 5 observations but these observations are picked by randomly resampling from original training data set. Individual classifiers are then trained on these data sets. Once the classifiers are trained, their classification performance can be evaluated by cross-validation, wherein they are evaluated on observations that they have not been trained on. This procedure gives an unbiased estimate of the classification error and is discussed in detail in section 2.2.1

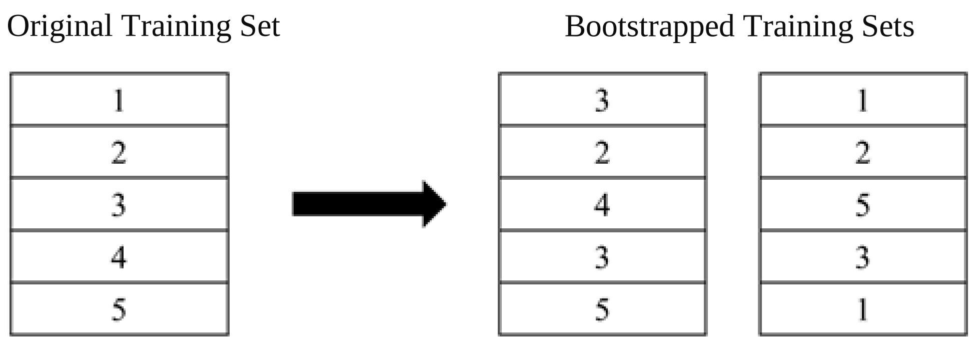

Figure 7 Bootstrapping.

The aim of bootstrapping is to increase the diversity of the training data. As suggested previously, increasing the diversity during training improves the generalizability when a suitable combination rule is applied. Diversity is achieved by using bootstrapped training data with different initial weights in the neural networks as well as using various classifiers (neural networks and support vector machines).

2.1.2 Classifiers

Once the bootstrapped training data have been generated, the classifiers need to be trained on these diversified data sets. Different classifiers are employed to overcome the problem of inductive bias. While, in this paper, a support vector machine and neural networks are employed; other classifiers can also be used in this fusion framework.

2.1.2.1 Least Square Support Vector Machine

Support vector machine (SVM) classification is based on Vapnik–Chervonenkis theory for structural risk minimization [27]. The objective of SVM is to find the optimal hyperplane $w^T x_i + b$ where w is the normal to the hyperplane and $|b|/\|w\|$ is the distance of the hyperplane from the origin of the coordinate system **Error! Reference source not found.**. Given an input feature vector $x_1, x_2, x_3, \dots., x_n$ where $x_i \in R^n$ and its corresponding class is $y_1, y_2, y_3, \dots., y_n$, where $y_i \in \{-1, 1\}$, the aim of the support vector machine is to find an optimal hyper plane w such that the objective function shown in Equation (10) is minimized subject to the constraints in Equation (11). To solve for the hyper plane, the following objective function is minimized which results in a hyperplane that optimally separates the two classes.

$$\min J(w, \varepsilon) = \frac{1}{2}\|w\|^2 + \gamma \sum_{i=1}^{N} \varepsilon_i{}^2 \qquad (10)$$

$$s.t. \qquad y_i(w^T \emptyset(x_i) + b) \geq 1 - \varepsilon_i \; when \; y_i \epsilon \{-1, 1\} \; and \; \varepsilon_i \geq 0. \qquad (11)$$

where ε_i is the slack variable or the distance margin introduced to allow misclassifications, and γ is the penalty or the cost for the misclassifications. The function $\emptyset(x_i)$ is a mapping of x_i to a higher dimensional space.

2.1.2.2 Neural Network

Given an input feature vector $x_1, x_2, x_3, \dots., x_K$, where $x_i \in R^n$ and its corresponding class $y_1, y_2, y_3, \dots., y_K \; \forall \; y_i \in \{-1, 1\}$, the neural network has *N* neurons in the input layer (where *N* is the size of input feature vector), one or more hidden layers and an output node [28]. A sigmoid activation function is used in each neuron in the hidden layer. The output of the neural network is the class label of the input features. We used a gradient-based approach to train the neural network without back propagation.

2.2 Fusion Parameter Computation

The diverse classifiers generated in the previous step need to be suitably combined. Here, a cost function has been formulated and minimized to obtain the most suitable combination of these classifiers. The fusion parameter computation includes two steps: performance evaluation and fusion optimization.

2.2.1 Performance Evaluation

To evaluate the classification performance, the bias and variance errors of each classifier need to be computed for the unseen observations by cross-validation. Bias and variance errors cannot be minimized simultaneously, because a reduction in either one of the two components could lead to an increase in the other, as shown in

Error! Reference source not found.. Therefore, the total error of a classifier is used to evaluate classification performance, which is a combination of both of these factors, as shown below [9]:

$$total\ error = \ bias^2 + variance \tag{12}$$

To estimate the bias of the classifiers, conventional validation methods will segment the training data into disjoint sets, e.g., training and validation data sets. For example, a data set can be segmented into two parts wherein 70% of the data are used to train the classifier and the remaining 30% are used for optimizing the classifier parameters [29]. But such a method is biased by the validation data. To choose how much data to use for training and validation, cross-validation has been thought to provide an unbiased estimate. In cross-validation, each of the trained classifiers is evaluated for accuracy on unseen observations.

2.2.1.1 Bias

If training data are resampled into B bootstrapped sample sets, each classifier is trained on different sample sets. Once all the classifiers are trained, the performance of each classifier is evaluated on the unseen observations in the training data. The errors computed are the false positive error (p_b) and false negative (n_b) error, as shown in (13).

$$p_b = \sum_{i=1}^{N} \frac{1}{|M_{bi}|} abs(\widehat{y_{bi}} - y_i) \qquad \forall\ x_i \nexists\ b\ and\ \ y_i = -1, b{:}\, 1 \rightarrow B \tag{13}$$

$$n_b = \sum_{i=1}^{N} \frac{1}{|M_{bi}|} abs(\widehat{y_{bi}} - y_i) \qquad \forall\ x_i \nexists\ b\ and\ \ y_i = 1\ b: 1 \rightarrow B$$

where $\widehat{y_{bi}}$ is the output of the classifier trained on the bootstrap sample set b; y_i is the actual class output with the input feature x_i; N is the total number of observations in the training data; and M_{bi} is the sample indices in the bootstrap sample set b for i^{th} observation $\forall\ x_i \nexists b$, and $|M_{bi}|$ is the number of such samples.

2.2.1.2 Variance

In ensemble learning, classifiers with high variance are susceptible to small changes in the input features. For example, neural networks that have too many hidden layers and nodes can have an over-fitting problem that results in high variance and low bias and therefore poor generalizability performance [30]. The variance of the classifier is given by equation (14):

$$Var_b = \sum_{i=1}^{N} (\widehat{y_{bi}} - E(\widehat{f(x_i)}))^2 \qquad \forall\ i: 1 \rightarrow N\ and\ x_i \nexists b \tag{14}$$

where $E(\widehat{f(x_i)})$ is the expected fusion outcome of the classifiers. Since a weighted fusion methodology will be employed, the expected value is given by the following equation:

$$E(\widehat{f(x)}) = \sum_{i=1}^{B} w_i \times \widehat{y_{bi}} \tag{15}$$

where $w_i$is the weight of the ith classifier.

2.3 Fusion

In this step, all the classifier model outcomes are combined with the weights computed by SQP optimization. The fusion output is computed by using a weighted sum of the classifier outputs, as shown in Equation (16). A signum function is used as a classification problem is being solved in this paper. The weighted sum of the outputs gives more priority to the classifiers that have shown better performance with the lowest cost of false positives and false negatives.

$$\widehat{f(x)} = sgn(\sum_{i=1}^{n} w_i \times \widehat{f(x)}_i) \tag{16}$$

The signum function (sgn) is used as decision threshold. This signum function is as defined in equations below.

$$sgn(x) = \begin{cases} -1 \; if \; x < 0 \\ 0 \; if \, x = 0 \\ 1 \; if \, x > 0 \end{cases}$$

Chapter 3: Objective Function Formulation and Optimization

This chapter summarized the steps involved in formulating the objective function and how this function is optimized. The primary objective of this fusion methodology is to equip users with a tool that is capable of optimally combining the results of different classification algorithms. The fusion optimization method helps achieve a balance between the bias–variance errors, thereby improving the classification accuracy and thus the generalizability.

3.1 Objective Function Formulation

The errors of false positive and false negative are calculated via cross-validation, as discussed in the previous section. For a given classifier H_b, let the false positive error be p_b and the false negative error be $q_b, b: 1 \rightarrow B$ (i.e., there are B classifiers trained on the bootstrapped data).

Let us assume that the cost of having a false positive is C_1 and that the cost for a false negative is C_2. The cost factors, C_1 and C_2, serve as prioritizing parameters, and they are relative terms that are used to prioritize the significance of false positives and false negatives in the cost function. For users who depend on system diagnostics for scheduling maintenance, the cost of a false positive is the cost incurred when a healthy system is erroneously classified as faulty, and the cost of a false negative is due to the erroneous classification of a faulty system as healthy. These costs may not necessarily be tangible; some of the intangible types of cost, such as safety, customer satisfaction, availability etc., could be incorporated. Quantifying these costs is out of the scope of this paper as these costs typically vary across organizations and applications. The total cost of the misclassification is as shown in Equation (17). The cost factors C_1 and C_2 can be changed based on the condition shown in Equation (18)

$$Cost\ of\ misclassification\ by\ a\ classifier\ (TC_b) = C_1 p_b + C_2 q_b \tag{17}$$

$$C_1 + C_2 = 1 \tag{18}$$

Now, we use a weighted sum of all the classifiers to obtain the final result. Let $w_1, w_2, \ldots, w_n$ be the weights assigned to each classifier. The objective function associated with the fusion of all these classifiers is as shown in (19). This objective function consists of two main components: the bias component and the variance component.

$$Objective\ function\ f(w) = \left(\sum_{b=1}^{B} (w_b \times TC_b)\right)^2 + \sum_{b=1}^{B} Var_b \tag{19}$$

where,

$$\sum_{b=1}^{B} w_b = 1 \tag{20}$$

$$0 < w_1, w_2, \cdots, w_B < 1 \tag{21}$$

3.2 Sequential Quadratic Programming

To minimize the above nonlinear Equation (19), nonlinear optimization techniques need to be used to find the optimal weights so as to reduce the objective function value. This problem of minimization is equivalent to a constrained nonlinear minimization [31]. The idea of using Sequential Quadratic Programming (SQP) is to find the optimal weights in B dimensional space [32]. SQP is an iterative quadratic programming–based approximation technique that is used to find the optimal values for the objective function in Equation (22) $\forall\ w \in R$ subject to equality and inequality constraints, where w is the set of weights $w_1, w_2, \cdots, w_B$. The non-linear problem in this case is the minimization of the objective function $f(W)$ subject to inequality and equality constraints in the vectors shown in Equations (23) and (24), respectively. The set of all points in the constrained space are referred to as the feasible set of the optimization problem. The algorithm can converge to an optimal state if the initial seed point is close to the optimal parameters' values. The initial seed point can move closer to the optimal solution in each iteration of the SQP computation.

$$\min(f(W)) \tag{22}$$

$$\text{s.t.} \qquad H(W) \leq 0 \tag{23}$$

$$G(W) = 0 \tag{24}$$

A slack variable z is introduced in Equation (23) such that $H(w) + z = 0$ and $z \geq 0$. The slack variable is introduced to convert the inequality constraint to an equality constraint. This is done for representation purpose, and later the slack variable is eliminated when the objective function is constructed. The Lagrangian for the minimization problem is given by Equation (25), where μ and σ are the set of Lagrangian multipliers associated with each of the constraints. This Lagrangian is sometimes referred to as the extended Lagrangian because it considers the slack variable z. The feasible solution W^* for the above constrained optimization problem needs to satisfy the first order necessary optimality conditions as shown in Equations (26)-(30) [33].

$$L\,(W, \mu, \sigma) = f(W) - \mu^T G - \sigma^T (H + z) \tag{25}$$

$$\nabla f(W^*) + \mu^{*T} \times \nabla G(W^*) + \sigma^{*T} \times \nabla H(W^*) = 0 \tag{26}$$

$$H(W^*) + z = 0 \; \forall \, i \, \in n \tag{27}$$

$$G(W^*) = 0 \; \forall \, i \, \in m \tag{28}$$

$$\mu^* \geq 0 \; \forall \, i \, \in m \tag{29}$$

$$z^{*T} \times \mu^* = 0 \tag{30}$$

The ∇ operator is used to define the gradient of a particular function. Hence $\nabla f(W^*)$ and $\nabla H(W^*)$ are the gradients of $f(W^*)$ and $H(W^*)$, respectively, at W^*. The optimization problem is obtained by solving a sequence of QP sub-problems, as shown in Equations (31)-(34) [33].

$$\frac{1}{2} d_W{}^T \times B(W_k) \times d_W + (\nabla L_k)^T \times d_W + (\nabla L_k)^T \times d_Z \tag{31}$$

$$\text{s.t.} \qquad \nabla H(W_k)^T \times d_W + H(W_k) = 0 \quad \forall\, i: 1 \rightarrow n \tag{32}$$

$$\nabla G(W_k)^T \times d_W + G(W_k) + d_z + Z_k = 0 \quad \forall\, i: 1 \rightarrow m \tag{33}$$

$$d_z \geq -Z_k{}^* \tag{34}$$

L_k is the extended Lagrangian at k^{th} iteration given by $L\ (W_k, \mu, \sigma)$. $B(W_k)$ in the above equation is the Hessian of the extended Lagrangian function L_k. W_k is the set of weights $\{w_1, w_2, \cdots, w_B\}$computed in the k^{th} iteration. d_W and d_z are the change vectors, i.e., they define the change of the variable W_k and Z_k. This QP sub-problem is a quadratic sub-problem with linearized constraints. This iteration is solved to obtain d_x and d_z for the k^{th} iteration. This can then be used to compute w_{k+1} and z_{k+1} for the next iteration $k+1$:

$$W_{k+1} = W_k + \alpha d_W \tag{35}$$

$$z_{k+1} = z_k + \alpha d_z \tag{36}$$

where α is the parameter used to ensure the convergence of the optimization problem. The SQP is an iterative process, where a sequence of feasible points W_k is computed such that they converge towards the optimal solution. The iteration can be terminated when the solution to the kth iteration meets the optimality conditions (26)-(30).

Chapter 4: Results and Data Analysis

The developed fusion method was used for diagnosing faults in different analog circuits. An analog circuit is termed faulty whenever, one or more critical components vary beyond their tolerance range. In this paper, two circuits are tested: a Sallen-Key band pass filter [34] as shown in Figure 8 and a biquad low pass filter as shown in Figure 10 [34]. The two case studies are discussed in the following two sub sections.

4.1Case Study 1: Sallen-Key Band Pass Filter

Its components, such as capacitors C and resistors R, have a tolerance range of 10%, which is obtained from the filter's data sheet specifications. The C_1, C_2, R_2 and R_3 are considered to be critical components in the filter since they can affect the frequency response of the circuit. The values of these components can be varied manually by using a variable capacitor and resistor. In this paper, eight seeded faulty conditions were introduced to test the proposed method, as summarized in Table 1.

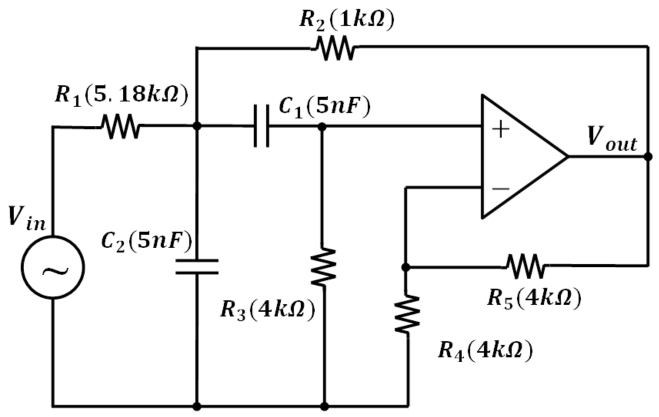

Figure 8 Sallen-Key band pass filter.

Table 1: Seeded Faulty Conditions

Fault Condition	**Nominal value**	**Sample Faulty Value**
C1 greater than nominal	5nF	6 nF,7 nF, 8 nF
C1 lesser than nominal	5nF	1.5 nF, 2.5 nF, 3.5 nF.
C2 greater than nominal	5nF	6 nF,7 nF, 8 nF
C2 lesser than nominal	5nF	1.5 nF, 2.5 nF, 3.5 nF.
R2 greater than nominal	1kΩ	1.24 kΩ, 1.61 kΩ, 2.21 kΩ
R2 lesser than nominal	1kΩ	372.3 Ω, 669 Ω, 836 Ω
R3 greater than nominal	2kΩ	2.31 kΩ, 2.53 kΩ, 3.66 kΩ
R3 lesser than nominal	2kΩ	0.883 kΩ, 1.5 kΩ, 1.79 kΩ

The circuit is defined as healthy when the critical component values vary within 10% of their nominal value. The condition of the circuit is defined as faulty when the components' parameter values change by greater or lesser than 10% of the nominal values.

The circuit was excited by using an input sweep signal of bandwidth 1–100 kHz for 100 ms. The output response of the circuit was then captured using an NI USB-6212 data acquisition board at a sampling rate of 200k samples/sec. The bandwidth of the sweep signal was larger than the operation frequency of the circuit. To obtain healthy features, a sweep signal was applied to the circuit with nominal values for the critical components for 150 times. Faulty features were obtained based on the faulty values in Table 1 for 50 times each. Hence, we obtained 50 × 3 = 150 responses for each fault condition. The total output responses, including healthy and faulty, are 150 + 50 × 3 × 8 = 1350.

Features were extracted from the output response using discrete wavelet transform. The response was decomposed into 5 decomposition levels. The energy calculated at each of these 5 decomposition levels represents our features.

To evaluate the performance of the proposed fusion method, the feature set was divided equally into two parts: one for training and the other for testing. Hence, the training and test data sets are each 675 × 5 in dimension. Both the training and test data sets consist of a healthy data set 75 × 5 in dimension and an unhealthy data set 600 × 5 in dimension. The test data was used to compare the performance of optimized fusion methodology with other techniques in the literature. The training data were resampled by a bootstrapping technique into 6 sample sets. The 6 bootstrap sample sets were used to train 6 classifiers, of which 3 were neural networks and the others were least square support vector machines.

A comparison of the optimized fusion methodology with others widely used fusion techniques and classifiers is shown in Table 2. It is observed that the fusion techniques outperformed single classifiers. False positives (false alarms) denote the alarms that were triggered even when there was no occurrence of a fault in the circuit; these are also referred to as false alarms. False negatives refer to situations where the alarms have not been triggered when there was an occurrence of fault in the circuit; they are also referred to as missed alarms. Considering field applications, where noise is likely to degrade the extracted features, an additive white Gaussian noise with a signal to noise ratio of 10dB was added to the input feature. The comparison results are shown in

Table **3**. It was observed that the optimized fusion approach provides more accurate diagnostic results in the presence of noise. The results were also compared when a limited amount of training data was used for training the classifiers, which is the common situation for many applications. In this case the classifiers were trained with a training set which is 270 × 5 in dimension. The training set contained healthy data set of 30 × 5 in dimension and unhealthy feature of 240 × 5 in dimension.

Table 2: Comparative Analysis

	Least Square Support Vector Machine	**Neural Network**	**Averaging**	**Localized weighted Fusion**	**Optimized Fusion**
False Positives (percentage)	2 (2.67%)	5 (6.67%)	2 (2.67%)	0 (0%)	0 (0%)
False Negatives (percentage)	0 (0%)	4 (0.67%)	3 (0.5%)	0 (0%)	0 (0%)

Table 3: Comparative Analysis with AWGN (SNR 10dB)

	Least Square Support Vector Machine	**Neural Network**	**Averaging**	**Localized weighted Fusion**	**Optimized Fusion**
False Positives (percentage)	15 (20%)	17 (22.67%)	13 (17.33%)	11 (14.66%)	6 (8%)
False Negatives (percentage)	16 (2.67%)	21 (3.5%)	11 (1.83%)	10 (1.66%)	3 (0.5%)

Table 4: Comparative Analysis with Truncated Training Data

	Least Square Support Vector Machine	Neural Network	Averaging	Localized Weighted Fusion	Optimized Fusion
False Positives (percentage)	14 (18.67%)	19 (25.34%)	15 (20%)	11 (14.66%)	4 (5.34%)
False Negatives (percentage)	33 (5.5%)	35 (5.83%)	33 (5.5%)	27 (4.5%)	14 (2.3%)

As can be seen from the Tables 3 and Table 4, lower percentages of false positives and negatives were observed in the proposed method compared to the other popular fusion algorithms, including average fusion [16][17] and localized fusion approach with bias compensation [23]. Apart from the comparison to the popular fusion techniques, the algorithm's performance was also compared with a single least square support vector machine (LS-SVM) classifier and a neural network classifier. The optimized fusion technique outperformed other classification techniques. In the current scenario, the cost of false positives and the cost of false negatives have been assumed to be the same. But for the users of diagnostic algorithms these costs could be altered to prioritize the reduction of either false positives or false negatives. For example, a higher cost factor for the false negative (C_2) compared to the cost of a false positive (C_1) would lead to prioritization of the reduction of the number of false negatives over false positives. Figure 9 below shows a change in the number of false positives and false negatives with the cost parameter C_1.

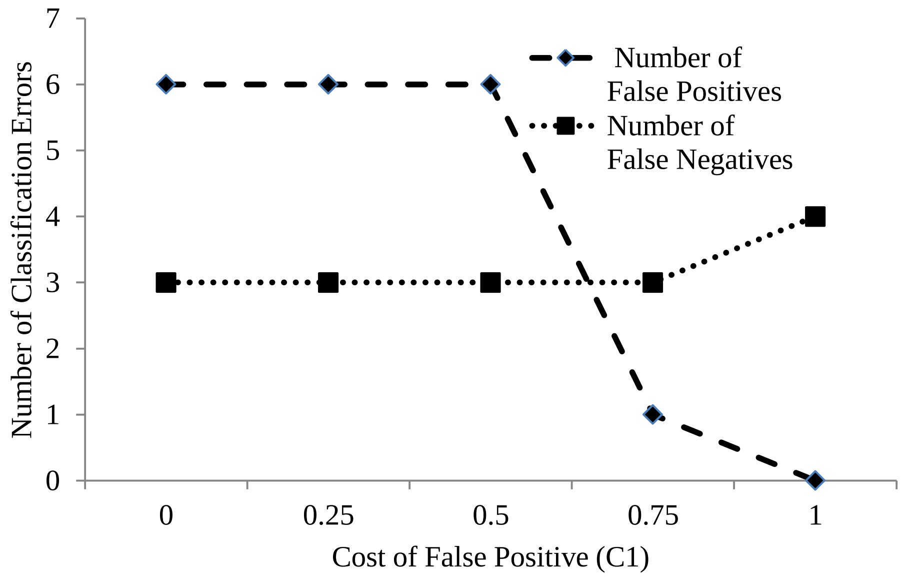

Figure 9 Change of false positive/ negative with cost parameter C1

4.2 Case Study 2: Biquad Low Pass Filter

Similar to Sallen-Key band pass filter this biquad low pass filter circuit also has critical components that affect the frequency response of the circuit. The critical components for the circuit were capacitors C1 and C2, and resistors R1, R2, R3 and R4 as shown in Figure 8. Table 5 summarizes the 12 seeded faulty conditions.

The data acquisition and feature extraction are similar to case 1, except we decomposed the output signal into 8 levels by using discrete wavelet transform. To obtain healthy features, a sweep signal was applied to the circuit with nominal values for the critical components for 50 times. Faulty features were obtained based on the faulty values in Table 5 for 10 times each. Hence, we obtained $10 \times 5 = 50$ responses for each fault condition. The total output responses, including healthy and faulty, are

50 + 10 × 5 × 12 = 650. The feature set was divided equally into two parts: one for training and the other for testing. Hence, the training and test data sets are each 325 × 8 in dimension. Both the training and test data sets consist of a healthy data set 25 × 8 in dimension and an unhealthy data set 300 × 8 in dimension.

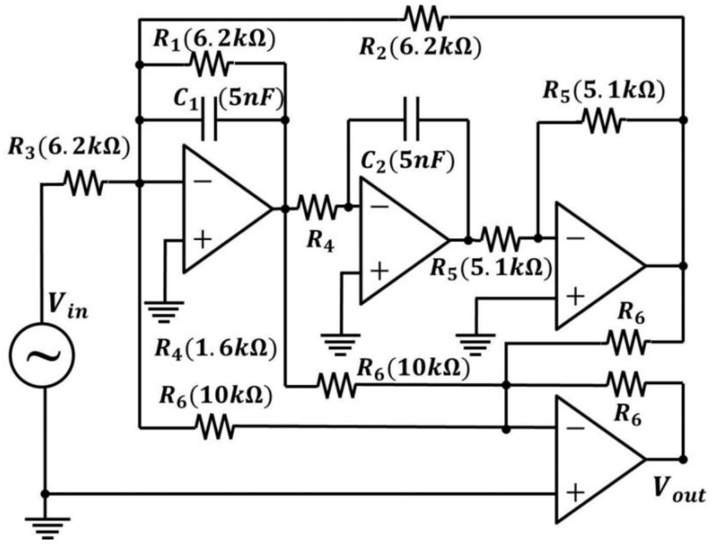

Figure 10 Biquad Low Pass Filter

Table 6 shows the comparative analysis on LS-SVM and fusion techniques. The results indicate a very similar performance for LS-SVM, localized weighted fusion and the optimized fusion technique. Averaging technique on the other hand had a very high false positive percentage. Table 7 shows the comparative analysis of the algorithms when additive white Gaussian noise with an SNR of 10dB was introduced in the extracted features. The results clearly demonstrate that the optimized fusion technique had a better performance when compared to other classification techniques. The performance of the algorithms was also analyzed with a truncated training data set of size 195 × 8. The training set contained healthy data set of 15 × 8 in dimension and unhealthy feature of 180 × 8 in dimension. Good performance in fault detection can be observed with a truncated data using an optimized fusion technique as shown in Table 8. From the graph shown in Figure 11 we observe that the cost parameters C_1 and C_2 can be used to prioritize the reduction in false positives and false negatives.

Table 5 Seeded Faulty Conditions

Fault Condition	**Nominal value**	**Sample Faulty Value**
C1 greater than nominal	5nF	6nF,7nF, 8nF, 9nF, 10nF
C1 lesser than nominal	5nF	4nF, 3.5nF, 3nF, 2.5nF, 2nF
C2 greater than nominal	5nF	6nF,7nF, 8nF, 9nF, 10nF
C2 lesser than nominal	5nF	4nF, 3.5nF, 3nF, 2.5nF, 2nF
R1 greater than nominal	6.2kΩ	7.4kΩ, 8.6kΩ, 10kΩ, 11.2kΩ,12kΩ
R1 lesser than nominal	6.2kΩ	5kΩ, 4.35kΩ, 3.72kΩ, 3kΩ, 2.5kΩ
R2 greater than nominal	6.2kΩ	7.4kΩ, 8.6kΩ, 10kΩ, 11.2kΩ,12kΩ
R2 lesser than nominal	6.2kΩ	5kΩ, 4.35kΩ, 3.72kΩ, 3kΩ, 2.5kΩ
R3 greater than nominal	6.2kΩ	7.4kΩ, 8.6kΩ, 10kΩ, 11.2kΩ,12kΩ
R3 lesser than nominal	6.2kΩ	5kΩ, 4.35kΩ, 3.72kΩ, 3kΩ, 2.5kΩ
R4 greater than nominal	1.6kΩ	7.4kΩ, 8.6kΩ, 10kΩ,

		11.2kΩ,12kΩ
R4 lesser than nominal	1.6kΩ	5kΩ, 4.35kΩ, 3.72kΩ, 3kΩ, 2.5kΩ

Table 6 Comparative Analysis

	Least Square Support Vector Machine	**Neural Network**	**Averaging**	**Localized weighted Fusion**	**Optimized Fusion**
False Positives (percentage)	0 (0%)	1 (4%)	2 (8%)	0 (0%)	0 (0%)
False Negatives (percentage)	0 (0%)	3 (1%)	1 (0.33%)	0 (0%)	0 (0%)

Table 7 Comparative Analysis with AWGN (SNR 10dB)

	Least Square Support Vector Machine	**Neural Network**	**Averaging**	**Localized weighted Fusion**	**Optimized Fusion**
False Positives	4 (16%)	4 (16%)	5 (20%)	3 (12%)	2 (8%)

(percentage)					
False Negatives (percentage)	21 (7%)	23 (2.67%)	23 (7.67%)	19 (6.34%)	10 (3.34%)

Table 8 Comparative Analysis with Truncated Training Data

	Least Square Support Vector Machine	**Neural Network**	**Averaging**	**Localized weighted Fusion**	**Optimized Fusion**
False Positives (percentage)	6 (24%)	9 (36%)	8 (32%)	5 (20%)	5 (20%)
False Negatives (percentage)	36 (12%)	43 (14.34%)	40 (13.34%)	33 (11%)	24 (8%)

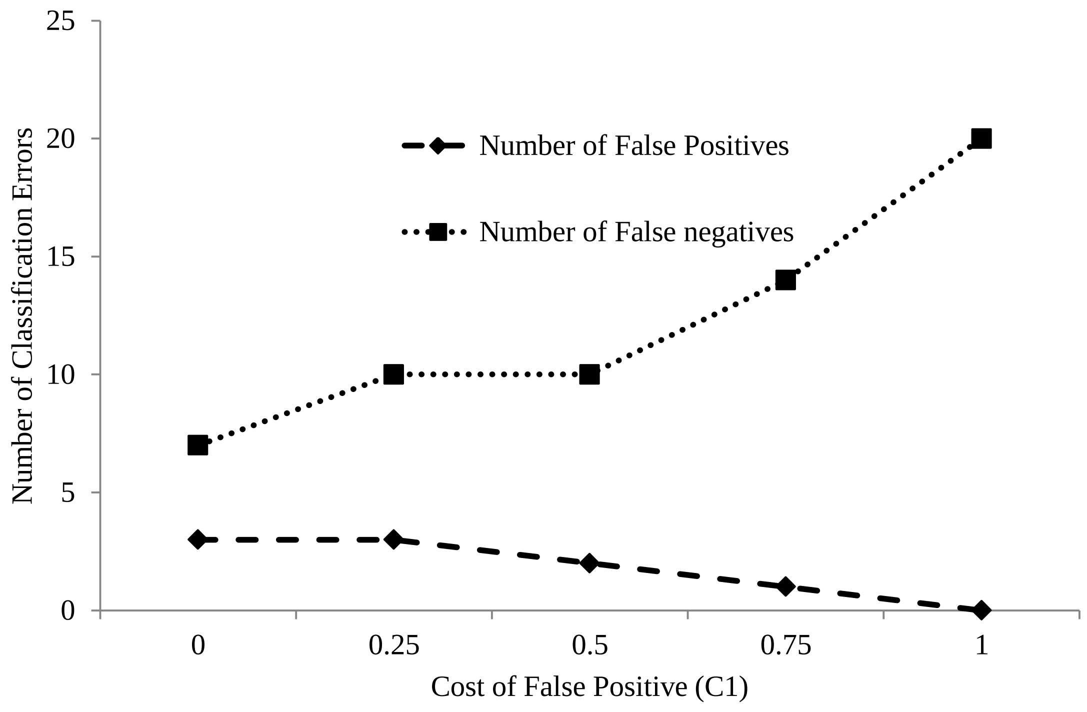

Figure 11 Change of False Positive/ Negative with Cost Factor C1

Chapter 5: Conclusion and Contributions

5.1 Conclusions

In this paper, we developed a combination rule to improve the classification accuracy and generalizability of diagnostic algorithms. Developed combination rule optimizes the balance between bias and variance using sequential quadratic programming. Cross validation was used to compute an unbiased estimate of the bias and variance of each classifier. The experimental results demonstrate that the developed fusion methodology has improved fault diagnostic accuracy over the averaging-based fusion technique, the localized fusion technique, and individual classifiers (neural network and support vector machine). Furthermore, developed algorithm incorporates the cost of false alarms and missed alarms in the learning algorithm for computing the combination rule. Taking the cost of false alarms and missed alarms into consideration helps in reducing the total cost misclassification the system as faulty or health. The performance of the algorithm was analyzed by varying the cost factors, and the results demonstrate that the cost factor used in the bias equation can prioritize either the false positives or the false negatives. The use of this combination rule is not limited by any particular classification algorithm. This method would enable the users of classification algorithms for diagnostics to input their costs (false positive cost and false negative cost) when choosing the most suitable combination rule for their application.

5.2 Contributions

This thesis developed a combination rule that optimizes the balance between bias and variance leading to improvement in classification accuracy compared to localized fusion and single classifiers. The bias variance balancing solves the bias variance dilemma in a classifier fusion problem.

Developed algorithm incorporates the cost of false alarms and missed alarms in the learning algorithm for computing the combination rule leading to reduction in the total cost of false alarms and missed alarms.

Bibliography

[1] Cheng. S, Azarian, M., and Pecht, M., "Sensor Systems for Prognostics and Health Management", Sensors, Vol. 10, pp.5774-5797, June, 2010.

[2] P.E.Utgoff, "Machine Learning of Inductive Bias", Kluwer Academic Publishers, 1986.

[3] Diana F. Gordon , Marie Des Jardins , G. Dietterich, "Evaluation and selection of biases in machine learning", ACM Computing Surveys, Vol. 5, pp. 255-306, 1995.

[4] Vapnik K., "The nature of statistical learning theory". Springer-Verlag, 1995.

[5] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Second edition, August 2009.

[6] Robi Polikar, "Ensemble based systems in decision making", IEEE Circuits and Systems Magazine, vol.6, no.3, pp.21-45, Third Quarter 2006

[7] A. Sharkey, Multi-Net Systems, Springer-Verlag, 1999.

[8] Gavin Brown, Jeremy Wyatt, Rachel Harris, Xin Yao, "Diversity Creation Methods: A Survey and Categorisation", Journal of Information Fusion, Vol. 6, No. 1, 2005.

[9] S. Geman, E. Bienenstock, R. Doursat, Neural networks and the bias/variance dilemma, Neural Computation 4 (1) (1992) 1–5.

[10] Oded Maimom and Lior Rokach, "Data Mining and Knowledge Discovery Handbook", 2nd edition, Springer, pp. 733-746, 2010.

[11] L. Breiman, “Bagging predictors,” Machine Learning, vol. 24, no. 2, pp. 123–140, 1996.

[12] R.E. Schapire, “The strength of weak learnability,” Machine Learning, vol. 5, no. 2, pp. 197–227, 1990.

[13] Arjun Chandra, Xin Yao, “Evolving hybrid ensembles of learning machines for better generalization”, Neurocomputing, vol. 69, pp. 686-700, 2006.

[14] Y. Liu, X. Yao, T. Higuchi, “Evolutionary ensembles with negative correlation learning”, IEEE Transactions on Evolutionary Computation, vol. 4, no. 4, pp. 380, 2000.

[15] H.A. Abbass, “Pareto neuro-evolution: constructing ensemble of neural networks using multi-objective optimization”, IEEE Conference on Evolutionary Computation, vol. 3, pp. 2074–2080, 2003.

[16] Hashem S. and Schmeiser B., Improving model accuracy using optimal linear combination of trained neural networks, IEEE Transactions on Neural Networks, 1995, Vol. 6, No. 3, 792-794.

[17] Xu L., Krzyzak A., Suen C Y, Methods of combining multiple classifiers and their application to handwritten recognition. IEEE Transactions on Systems, Man, and Cybernetics Vol. 22, No. 3, pp 418-435, 1992.

[18] L.I. Kuncheva, “Combining Pattern Classifiers, Methods and Algorithms. New York, NY: Wiley Interscience 2005.

[19] Kimura F, Shridhar M., Handwritten Numerical Recognition based on Multiple Algorithms, Pattern Recognition, Vol. 24 No. 10, 969-983.

[20] Franke J, Mandler E., "A Comparison of two Approaches for combining the votes of cooperating classifiers, Proceesings of 11th international conference on pattern recognition, Vol. 2, pp 611-614, 1992.

[21] N. Littelestone and M. Warmuth, "The Weighted Majority Algorithm", Information and Computation, Volume 108, Issue 2, Pages 212–261, 1994.

[22] W. Philip Kegelmeyer Jr. and Kevin Bowyer, "Combination of Multiple Classifiers Using Local Accuracy Estimates", IEEE Transactions on Pattern Analysis and Machine Learning, Vol. 19, no. 4, pp 405-410, 1997.

[23] Feng Xue, Raj Subbu, Piero Bonissone, "Locally Weighted Fusion of Multiple Predictive Models", International Joint Conference on Neural Networks, 2006.

[24] Piero P. Bonissone, Feng Xue, Raj Subbu, "Fast meta-models for local fusion of multiple predictive models", Applied Soft Computing, Volume 11, Issue 2, March 2011, Pages 1529–1539.

[25] Stéphane Tufféry, "Data Mining and Statistics for Decision Making", Wiley; 2nd edition, 2011.

[26] Efron, B., "Bootstrap Methods: Another look at the Jackknife", Annals of Statistics, Vol. 7, No1, pp 1-26, 1979.

[27] Vapnik K., "The nature of statistical learning theory". Springer-Verlag, 1995.

[28] Kevin Gurney, Kevin N. Gurney, "An Introduction to Neural Networks", CRC Press, 1997.

[29] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Second edition, August 2009.

[30] Erica Briscoe, Jacob Feldman, "Conceptual complexity and the bias/variance tradeoff", Cognition Vol. 118 pp. 2–16, 2011.

[31] Gill, PhilipE. and Wong, Elizabeth, "Sequential Quadratic Programming Methods", Mixed Integer Nonlinear Programming in The IMA Volumes in Mathematics and its Applications, pp 147-224, Springer 2012.

[32] W. Forst and D. Hoffmann, "Optimization—Theory and Practice", Springer, 2010.

[33] Byrd, R. H, J. C Gilbert and J. Nocedal, "A trust region method based on interior point techniques for non-linear programming", Mathematical Programming, Vol.89, No.1, pp149-185, 2000.

[34] Arvind Sai Sarathi Vasan, Bing Long and Michael Pecht, Diagnostics and Prognostics Method for Analog Electronic Circuits, IEEE Transactions on Industrial Electronics, Vol. 60, 2012.