Metrol. Meas. Syst., Vol. XVII (2010), No. 4, pp. 561–582

PAN POLSKA AKADEMIA NAUK

METROLOGY AND MEASUREMENT SYSTEMS

Index 330930, ISSN 0860-8229

www.metrology.pg.gda.pl

M&MS

A NOVEL APPROACH OF ANALOG FAULT CLASSIFICATION USING A SUPPORT VECTOR MACHINES CLASSIFIER

Jiang Cui, Youren Wang

College of Automation Engineering, Nanjing University of Aeronautics and Astronautcs, Nanjing City, Jiangsu Province, China
(✉ cuijiang@nuaa.edu.cn, +86 137 7065 3061)

Abstract

In order to make the analog fault classification more accurate, we present a method based on the Support Vector Machines Classifier (SVC) with wavelet packet decomposition (WPD) as a preprocessor. In this paper, the conventional one-against-rest SVC is resorted to perform a multi-class classification task because this classifier is simple in terms of training and testing. However, this SVC needs all decision functions to classify the query sample. In our study, this classifier is improved to make the fault classification task more fast and efficient. Also, in order to reduce the size of the feature samples, the wavelet packet analysis is employed. In our investigations, the wavelet analysis can be used as a tool of feature extractor or noise filter and this preprocessor can improve the fault classification resolution of the analog circuits. Moreover, our investigation illustrates that the SVC can be applicable to the domain of analog fault classification and this novel classifier can be viewed as an alternative for the back-propagation (BP) neural network classifier.

Keywords: analog circuits, fault classification, Support Vector Machines Classifier, Neural Networks, wavelet packet decomposition.

1. Introduction

For nearly forty years, the subject of analog circuit fault diagnosis has been of interest to researchers in the domain of analog testing. The fault diagnosis can be divided into two parts: fault detection and fault localization [1]. Fault detection technique can detect whether a circuit under test is faulty. Next, the fault localization technique is employed to find which component or sub-system module is faulty. In our conception, the fault diagnosis can be summarized to the problem of fault classification. Fault classification can be performed by a circuit mathematical model or an artificial intelligence approach. The circuit model can be used to perform fault location or even parameter identification easily, but, in fact, it is hard to obtain an accurate model even for a linear analog circuit because of component value variation (*i.e.* tolerance) resulting from manufacturing technology or other factors. Also, it is complicated to establish an analog fault model because the number of possible analog faults can be infinite. Another problem is the limitation of accessible nodes of the analog circuit, and this limitation will probably impair the diagnosis resolution. Focusing on these difficulties of analog fault classification, artificial intelligence seems to be the most effective tool in analog circuit fault classification. In the past decades, the Neural Networks (NNs) based methods has been applauded by researchers in diagnosing analog linear circuits or even nonlinear circuits[2, 3]. The NN can learn the samples itself according to some training rules and after a training stage it can predict a sample which does not belong to the training samples. A large number of articles have addressed the applications of neural networks in analog circuit diagnosis[2–9] and a review of this literature indicates the importance of NNs in the application to analog fault diagnosis.

Article history: received on Jul. 29, 2010; accepted on Oct. 26, 2010; available online on Nov. 26, 2010.

Up to now, the back-propagation neural network (BPNN) was the most popular classifier in the analog diagnosis domain, but this ANN still faces some difficulties, *e.g.*, easy entrapment into the local minima during the training stage, long training time to convergence *etc*. In order to solve these problems, some additional measures or even different neural networks have to be considered. For example, the genetic algorithm (GA) is used in [5] for finding a global minimal solution; in [4], the learning vector quantization neural network (LVQNN) is adopted to avoid this local minimum in training stage; a modular diagnostic-system is used in [6] to replace the single neural network with many small-sized neural networks, and this replacement can give a flexible diagnosis of the circuit at component level or even system level. Also, the ANN is sensitive to the data dimension of the training samples. High-dimensional data always results in a long training time, and sometimes, failure to converge. Hence, a proper preprocessor is necessary. In the application of analog circuit diagnosis with NNs, the widely used preprocessor is the wavelet decomposition technique [3, 5–8]. The wavelet decomposition is a multi-resolution analysis method which can get the details and approximations (coefficients) of the signal.

In this paper, the wavelet packet decomposition technique is utilized to reduce the feature size, then, a fault classifier is designed to perform fault classification (including detection and localization). The presented classifier in this paper is a multi-class SVC, which is based on an ensemble of binary support vector machines classifiers (BSVC). The SVC is characterized by fast convergence to the global optimization, excellent generalization capability and immunity to high-dimensional data, *etc*. These characteristics make the SVC an attractive classifier in diagnosing the analog circuits and the consequent experiments also prove the SVC is applicable to analog circuit diagnosis. In the past several years, some researchers have begun to use the SVC to perform the analog circuit diagnosis task [10–12]. The frequently used SVC is based on the structure of so-called one-against-one or the one-against-rest. In [12], the author employs a multi-class SVC which has a one-against-rest structure to perform fault classification task. In [10] and [11], the one-against-one SVC is preferred for this task. For N fault classes, a one-against-one SVC has to train N (N–1) / 2 BSVCs and in the diagnosis stage, N (N–1) / 2 decision functions must be calculated. In our study, a one-against-rest SVC is resorted to perform analog circuit fault isolation because this classifier has a simple structure compared to the one-against-one SVC. For N fault classes, the conventional one-against-rest SVC will train N BSVCs, and thus, only N calculations of all decision functions are needed. In this paper, the conventional one-against-rest SVC is further improved and this improvement will contribute to the reduction of testing time while keeping diagnosis accuracy acceptable.

This paper is organized in the following order. In Section 2, we give a concise introduction to BSVC and several multi-class SVCs are also outlined in this section. The proposed SVC as well as the fault decision algorithm is discussed in Section 3. In Section 4, the method is validated by the experiment results from simulated circuits as well as actual circuits with discrete components. Results based on several tables and figures are given in Section 5. Useful conclusions are presented in Section 6.

2. Multi-class support vector machines classifiers used in this study

2.1. Binary support vector machines classifier (BSVC)

For binary classification, let $\{(x_i, y_i)\}$ $(i = 1, 2, ..., Q)$ be a set of training samples. Each sample $x_i \in R^d$, d being the dimension of the input space, is assigned to $y_i \in \{+1, -1\}$. The input space is mapped via the mapping function $\phi(\cdot)$ to a high-dimensional linear space, where an

optimal hyper-plane (W^*, b^*) is found to separate the sample x with indicator function $sign(f(x))$:

$$sign(f(x)) = \begin{cases} +1 & f(x) \geq 0 \\ -1 & f(x) < 0 \end{cases},$$

where:

$$f(x) = W^* \cdot \phi(x) + b^* \tag{1}$$

In the case of nonlinear separable training samples, slack variables $\xi_i \geq 0$ are introduced. Considering the criterion of maximal margin/error minimization leads to the following optimization problem:

$$\begin{aligned} &\psi(W) = \|W\|^2 \Big/ 2 + C\sum_{i=1}^{Q} \xi_i \\ &s.t. \qquad y_i(W \cdot \phi(x_i) + b) - 1 + \xi_i \geq 0 \end{aligned} \tag{2}$$

where C is the upper bound, controlling the tradeoff of the classification boundary complexity and classification error. Solving this optimization problem will lead to a quadratic program (QP) solution, in which Lagrange Multipliers λ_k are introduced:

$$f(x) = \sum_{k=1}^{n_{sv}} y_k \lambda_k K(x, x_k) + b^*, \tag{3}$$

where n_{sv} is the number of total support vectors, $\lambda_k > 0$ is the Lagrange multiplier of the k^{th} support vector, and $K(x, x_k) = <\phi(x), \phi(x_k)>$ is the kernel function, here, $<\cdot,\cdot>$ is the dot product.

Generally, the support vectors can be divided into two types: unbounded support vectors (UBSV) and bounded support vectors (BSV). The UBSV refers to the support vectors whose corresponding Lagrange multipliers are less than C and the BSV refers to the support vectors whose Lagrange multipliers are equal to C. Here, the kernel function must meet Mercer's condition [13]. In our study, both the q^{th}-order polynomial kernel function and the radial basis kernel (RBF) function are used:

$$K(x, y) = (1 + x^T \cdot y)^q, \tag{4.1}$$

$$K(x, y) = e^{\frac{-|x-y|^2}{\sigma^2}}, \tag{4.2}$$

where σ is the width of the kernel function and the superscript T is the transpose of column vector x.

2.2. Multi-class SVC

Many practical problems, such as analog circuit diagnosis, fall into the category of multi-class classification. In order to solve a multi-class problem, several BSVCs must be combined, or even a new multi-class support vector classifier should be considered [14]. In our research, two typical multi-class SVC techniques are used and compared.

The first one is the one-against-rest SVC, which was invented by Vapnik [15]. In the training stage, N BSVCs are constructed for N classes, and for each training process, the

i^{th} class (represented with "-1" label in our study) is separated from the other (N-1) classes (represented with "+1" label in our study). In the decision stage, in order to test query sample x, the Winner-Takes-All (WTA) rule is always adopted for all the decision functions. Let $f_i(x)\,(i=1,...,N)$ be the decision function of the i^{th} BSVC:

$$f_i(x)=\sum_{k=1}^{n_{sv}^i} y_k^i\,\lambda_k^i\,K(x,x_k^i)+b_i^*, \tag{5}$$

where n_{sv}^i is the support vector number for the i^{th} function $f_i(x)$, λ_k^i is the Lagrange multiplier of the k^{th} support vector, y_k^i is the label of the k^{th} support vector, and b_i^* is the bias of $f_i(x)$.

The WTA rule is:

$$\arg\min_{i=1,2,...,N}(f_i(x)), \tag{6}$$

The second conventional multi-class method is the one-against-one SVC. In the training stage, all possible pair classes are trained and altogether N (N–1) / 2 BSVCs are constructed. In the decision stage, Max-wins strategy is adopted. In this paper, we use the decision method described in [14].

Other types of multi-class SVC, such as the decision-tree or hierarchical SVC [16], will probably generate different classifier structures by different combinations of BSVCs. This means that additional measures need to be adopted to gain a viable classifier structure, and these measures are usually complex and time-consuming. In our study, these methods are not addressed.

3. The proposed fault classification method

3.1. Method principle

The SVC proposed in our experiments is based on the one-against-rest SVC. The conventional one-against-rest SVC requires the calculation of all the decision functions, which are not necessary for most of diagnosis cases.

For instance, suppose to classify three fault classes (see Fig. 1), represented by "1", "2" and "3" (in the high-dimensional space) respectively.

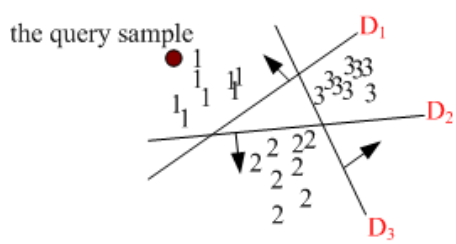


Fig. 1. Three classes are separated by the one-against-rest SVC.

In Fig. 1, the one-against-rest SVC is designed to generate three BSVCs, whose optimal decision hyper-planes are D_1, D_2 and D_3 respectively. In this illustration, the arrow direction of the optimal hype-plane indicates the label of the training sample (*e.g.*, the arrow direction of D_1 indicates the label of class "1" is -1 and the other classes are represented by label +1). Assume the query sample x to fall into the area of class "1", then the decision functions for the sample should be : $f_1(x)<0$, $f_2(x)>0$ and $f_3(x)>0$. In other words, the query sample

can be easily assigned to class "1" from the polarity of the decision function outputs. In this case, it is the decision function $f_1(x)$ that gives the classification information and the other two decision functions ($f_2(x)$ and $f_3(x)$) are redundant.

A special case is that the query sample falls into the unclassifiable region (UR, a public region formed by more than two hyper-planes, whose decision function outputs are all informative) or the rejected region (RR, a public region formed by all hyper-planes), as shown in Fig. 2. This phenomenon occurs when two or more fault classes becomes overlapped in the measurement space. In our investigations, soft classes are easy to overlap in the measurement space because the overlapped soft classes always have some similar samples.

In the case of UR, the query sample falls into the public area formed by D_1 and D_3. From the figure, it is easy to get $f_1(x)<0$, $f_2(x)>0$ and $f_3(x)<0$. Obviously, both $f_1(x)$ and $f_3(x)$ are informative, and $f_2(x)$ is redundant.

If the sample falls into the RR, which is formed by all the decision hyper-planes, then we get $f_1(x)>0$, $f_2(x)>0$ and $f_3(x)>0$. This case means that none of the decision functions is informative.

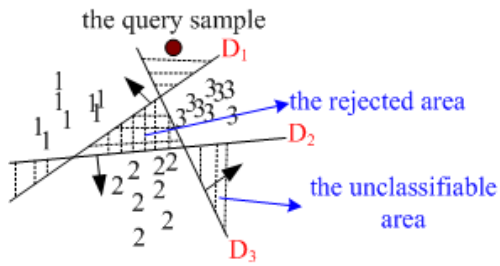


Fig. 2. The query sample falls into the UR formed by D_1 and D_3. In this figure, all URs are shaded by the dashed-lines and the RR is shaded by crossed dashed-lines.

In the application of the one-against-rest SVC to analog circuit fault classification, if we know which decision functions are informative, many computations will be avoided and the testing time will be expected to reduce remarkably. This technique will be useful in analog fault detection and localization.

3.2. Fault decision algorithm

Our fault decision algorithm is based on the number of informative decision functions (NIDF). Different NIDFs will lead to different fault decision algorithms.

– NIDF=1. This means only one decision function ($f_i(x)<0$) is informative and in this case, the query sample should be assigned to the i^{th} fault class.
– NIDF>1. In this case, the sample falls into the so-called UR. We must take measures to deal with this case, or the classifier can not decide on the assignment of the sample. Our method is based on a heuristic assumption that the sample has a closer space distance to the fault class it should be assigned to. In our method, we calculate the space distances between the query sample and the hyper-planes involved in decision calculations. Our method is simply illustrated with Fig. 3.

Assume the query sample to fall into the UR formed by D_1 and D_3. We calculate the space distance d_1 and d_3, shown in Fig. 3, respectively. If $d_3 < d_1$, we assign the query sample to fault class "1", otherwise, the query sample will be assigned to class "3". This heuristic method works well in our experiments. Because, from the point view of space distance, the

larger the distance is, the bigger the possibility with which the sample belongs to the corresponding class is.

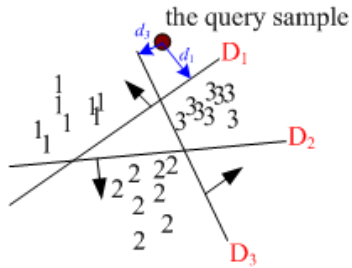


Fig. 3. Space distance based method is taken to resolve the UR.

The space distance d_i is calculated in the high-dimensional space (*i.e.* kernel space), onto which the input samples are mapped via the nonlinear mapping function $\phi(\square)$:

$$d_i(x) = \frac{|f_i(x)|}{\|\mathrm{W}_i\|}, \tag{7}$$

where x is the query sample, f_i is the decision function of hyper-plane D_i, $\|.\|$ is the 2-norm of the weight vector W_i of D_i.

According to the principle of support vector machines, W_i can be calculated by:

$$\mathrm{W}_i = \sum_{k=1}^{n_{sv}^i} y_k^i \lambda_k^i \phi(x_k^i). \tag{8}$$

It is the UBSVs that mainly determine the classifier performance, and then, only the UBSVs are considered in our algorithm. Hence, (8) is changed to be:

$$\mathrm{W}_i = \sum_{l=1}^{n_{ubsv}^i} y_l^i \lambda_l^i \phi(x_l^i), \tag{9}$$

where n_{ubsv}^i is the number of UBSV of the i^{th} decision function.

Also, $\|\mathrm{W}_i\|$ can be computed with the dot product form:

$$\|\mathrm{W}_i\| = \sqrt{<\mathrm{W}_i, \mathrm{W}_i>}. \tag{10}$$

Considering the kernel function principle $K(x,y) = <\phi(x), \phi(y)>$, $\|\mathrm{W}_i\|$ turns out to be:

$$\begin{aligned} \|\mathrm{W}_i\| &= \sqrt{\sum_{j=1}^{n_{ubsv}^i} \sum_{l=1}^{n_{ubsv}^i} y_j^i y_l^i \lambda_j^i \lambda_l^i <\phi(x_j^i), \phi(x_l^i)>} \\ &= \sqrt{\sum_{j=1}^{n_{ubsv}^i} \sum_{l=1}^{n_{ubsv}^i} y_j^i y_l^i \lambda_j^i \lambda_l^i K(x_j^i, x_l^i)} \end{aligned} \tag{11}$$

In this case, our decision algorithm becomes:

$$x \in \arg(\max_i(d_i)). \tag{12}$$

– NIDF=0. In this case, the sample falls into the RR, which is shown in Fig. 4.

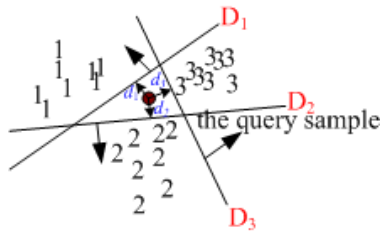


Fig. 4. Space distance based method is taken to resolve the RR

We still adopt the space distance measure to decide on the assignment of the query sample. In Fig. 4, the space distances between the sample and all hyper-planes are computed. Our fault decision algorithm depends on the following rule:

$$x \in \arg(\min_{i}(d_i)) . \tag{13}$$

This decision rule indicates that the smaller the distance is, the bigger the possibility with which the sample belongs to the corresponding fault class on the other side of the decision hyper-plane. For instance, in Fig. 4, if $d1 \le d2 \le d3$, then, according to (13), the query sample should be assigned to class "1".

3.3. Method implementation

In fact, it is quite difficult to directly get the informative decision functions with an accurate and fast method. The most reliable method is to calculate all decision functions one by one and then, select these informative functions directly. But, in this general operation, many redundant decision functions are also involved and too much time will be consumed on these redundant decision functions. In this paper, we use the Euclidean distance in measurement space to obtain the informative decision functions. Euclidean distance calculation is well known in analog fault dictionary (FD) applications, and this method is not accurate but very fast. In our study, this distance based method can give one or more decision function candidates, based on which our method is performed.

Prior to the use of our approach, three FDs need to be constructed. The first FD (FD_1) is very simple, and it only contains the centroids of all fault classes. Suppose to classify N fault classes and each class contains M training samples. The centroids are defined as below:

$$C_j = \frac{1}{M}\sum_{i=1}^{M} x_{ij} , \tag{14}$$

where x_{ij} is the i^{th} training sample of fault class j ($j = 1,...,N$).

The second FD (FD_2) contains the training parameters of the one-against-rest SVC. In our design, every BSVC is trained and the corresponding parameters (such as the support vectors, Lagrange multipliers, bias, kernel function type and the related kernel parameters) are saved. For convenience of expression, let BSVC_j ($j = 1,...,N$) be the BSVC to separate fault class j from the remaining fault classes.

The third FD (FD_3) contains the $\|W_i\|$ (or $1/\|W_i\|$, $i=1,...,N$). This FD will make the calculation more fast in case the sample falls into the RR.

Our method can be divided into three steps and these steps are illustrated clearly with the flow chart shown in Fig. 5.

In the first step, the Euclidean distance Ed_j ($j=1,...,N$) between the query sample x and centroid C_j is calculated respectively. This calculation is not accurate to decide on the informative decision function directly, but it is simple and very fast. In our investigation, the time consumed for this calculation can be negligible when compared to the next steps. In order to find the possible candidates of informative decision functions, the sort operation is needed and the indices corresponding to the fault classes also need to be saved to a variable index [*N*]. In this step, the first FD is used.

In the second step, the candidate decision functions are calculated one by one via FD_2. The signal function is employed to decide on the informative functions. In our design, once an informative decision function is found, the next loop is still executed until a redundant one is found. This arrangement can try to avoid the loss of informative decision functions. For instance, if the output of $BSVC_1$ is negative while the output of $BSVC_2$ is positive, then the informative BSVC should be $BSVC_1$ and so, the program will turn to the third stage for final fault detection or localization.

In the third step, the program can easily determine the region the sample falls into and hence the corresponding fault decision algorithm is adopted. The time needed in this step can be negligible, because in this step, both of $f_i(x)$ (calculated from the second step) and FD_3 (*i.e.* $1/\|W_i\|$) are already available. In this step, FD_3 can be used depending on the NIDF.

3.4. Computational complexity for the proposed SVC

According to the flow chart, the computational complexity will contain three parts as follows. Let I be the dimension of the samples, $\overline{n_{sv}}$ the average of the support vectors of the one-against-rest SVC, n_{BSVC} the number of BSVCs involved in the second step, n_{iBSVC} the number of informative decision functions.

- In the first step, the computational load is mainly from the sorting operation, whose complexity is $O(I \cdot N \log_2^N)$.
- In the second step, the computational complexity is mainly from the calculation of decision functions. According to [17], the calculation cost of a single BSVC decision function is $O(H \cdot \overline{n_{sv}})$, where $H = O(I)$ is the computational cost of kernel function (polynomial kernel or RBF kernel). Hence, this step will require a calculation cost of $O(n_{BSVC} \cdot H \cdot \overline{n_{sv}})$.
- In the third step, the decision conclusion can be drawn from all informative decision functions, so, the computational complexity should be $O(n_{iBSVC})$.

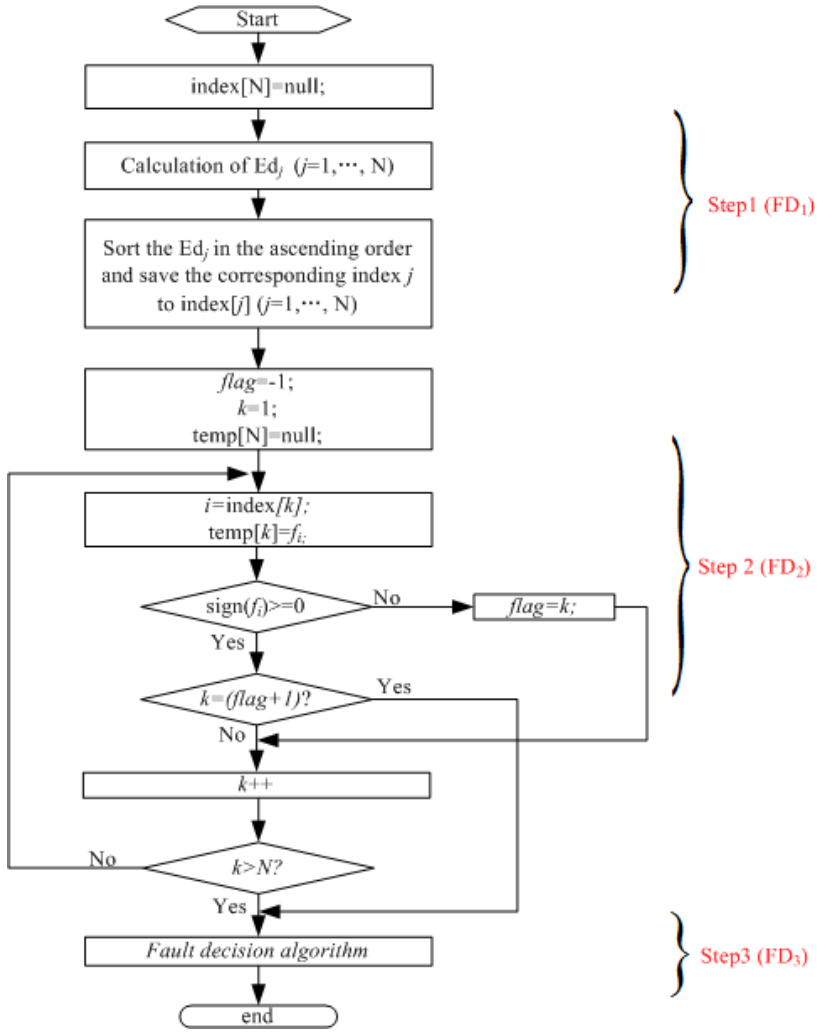


Fig. 5. Implementation flow chart of the proposed method.

Hence, the total complexity O_{total} of the proposed method should be:

$$\begin{aligned} O_{total} &= O(\mathrm{I}\cdot N\cdot \log_2{}^{N}) + O(n_{BSVC}\cdot H\cdot \overline{n_{sv}}) + O(n_{iBSVC}) \\ &= O(\max\{\mathrm{I}\cdot N\log_2{}^{N}, n_{BSVC}\cdot H\cdot \overline{n_{sv}}, n_{iBSVC}\}). \end{aligned} \tag{15}$$

For a practical problem, generally, $n_{BSVC} < N$, $n_{iBSVC} < N$ and $N < 2^{\overline{n_{sv}}}$, thus we have:

$$O_{total} = O(n_{BSVC}\cdot H\cdot \overline{n_{sv}}) < O(N\cdot H\cdot \overline{n_{sv}}). \tag{16}$$

In a word, the computational cost of the proposed method is smaller than that of the conventional one-against-rest SVC, whose computational complexity can be expressed with (16).

4. Analog circuits used and feature extraction

4.1. Linear Circuits

The first circuit under test (CUT), as shown in Fig. 6, is a Sallen-Key band pass filter (BPF) [6], [12] with 24.5 KHz central frequency.

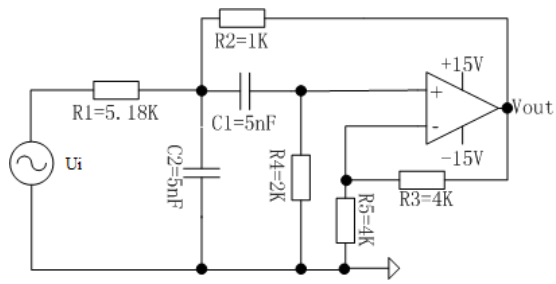


Fig. 6. First analog circuit under test.

In this experiment, the operational amplifier output node Vout is accessible. For this CUT, a fault-free component has a model of $X(1\pm k)$, where X is the nominal value, and k is the tolerance. In this paper, k is set at 5% for resistors and 10% for capacitors. A soft fault model of component value lower (higher) than the nominal value is expressed with $[X*(1-f), X*(1-k)]$ ($[X*(1+k), X*(1+f)]$) respectively, where f is the fault tolerance. In this paper, f is set at 50%. The fault classes used in this circuit are listed in the following order: nf, R2↓, R2↑, R4↓, R4↑, C1↓, C1↑, C2↓, C2↑, where "↑" ("↓") indicates the component value is higher (lower) than the nominal value. Also, the fault class "nf" means the CUT is fault-free.

The second filter is a High-Pass filter [8], which is shown in Fig.7. For this circuit, Vout is the only accessible node. In this experiment, the faulty components and the fault classes are all listed in order in Table 1. The component tolerance listed in this Table is for future software simulation with Monte Carlo analysis.

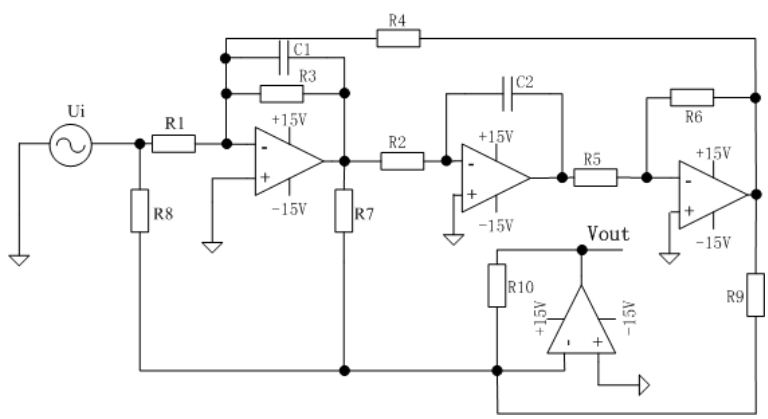


Fig. 7. Second analog circuit under test.

For both circuits simulated with SPICE software, the stimulus is a pulse with 10 μs duration and 5 V peak. For every fault class, the samples are generated by varying the circuit

faulty components within their nominal tolerances while the faulty component value is set to a fault value or changes evenly within its fault tolerance

Table 1. Fault classes designed for the second CUT. Nominal and faulty components values are also specified.

Order	Fault class	Nominal value	Tolerance	Fault value
1	nf	-	-	-
2	C1↑	5nf	10%	10nf
3	C1↓	5nf	10%	2.5nf
4	R4↑	1600 Ω	5%	2500Ω
5	R4↓	1600Ω	5%	500Ω
6	C2↑	5nf	10%	15nf
7	C2↓	5nf	10%	1.5nf
8	R3↑	6200Ω	5%	12000Ω
9	R3↓	6200Ω	5%	2700Ω
10	R2↑	6200Ω	5%	18000Ω
11	R2↓	6200Ω	5%	2000Ω
12	R1↑	6200Ω	5%	15000Ω
13	R1↓	6200Ω	5%	3000Ω

4.2 Nonlinear circuits

The first nonlinear analog circuit is a differential amplifier with Q2N2222 transistors (Q1 and Q2, shown in Fig. 8) and in this circuit, Q3 and Q4 form a basic current mirror. In this study, a 10 Hz sine wave signal with 0.05 V is used to excite the circuit and the responses are collected via the collector of Q2 (*i.e*. Vout).

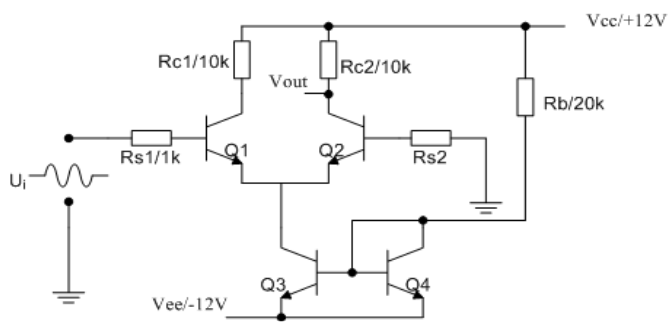


Fig. 8. A nonlinear circuit.

In this analog circuit, single soft faults for Rc1, Rc2 and Rb are considered. The soft faults are summarized in Table 2. For the resistors, the soft fault model increasing the value is designed as $[X+6\sigma, X+10\sigma]$, where σ indicates a sigma variation from the nominal value X.

For the transistors (Q1~Q4), single hard faults (open or short) are also under consideration. In this study, the hard fault for a transistor can have six cases as illustrated in Fig. 9. The open fault for single terminal of the transistor is simulated by adding a 100 Mohm resistor in series with this terminal (the open fault classes are QBo, QCo and QEo corresponding to (a), (b) and (c) respectively in Fig. 9; the short fault is simulated by adding a 1 ohm resistor in parallel with these two terminals (the short fault classes are QBCs, QCEs and QBEs corresponding to (d), (e) and (f) respectively in Fig. 9).

Table 2. Single faults selected for the differential amplifier.

Fault class	Nominal value	Tolerance	Fault model
Rc1↑	10k	5%	$[X+6\sigma, X+10\sigma]$
Rc1↓	10k	5%	$[X-10\sigma, X-6\sigma]$
Rc2↑	10k	5%	$[X+6\sigma, X+10\sigma]$
Rc2↓	10k	5%	$[X-10\sigma, X-6\sigma]$
Rb↑	20k	5%	$[X+6\sigma, X+10\sigma]$
Rb↓	20k	5%	$[X-10\sigma, X-6\sigma]$

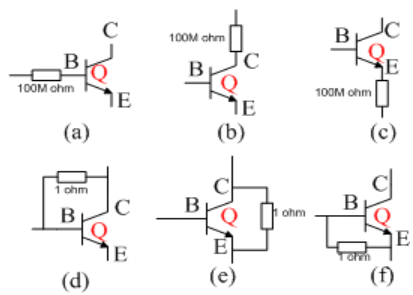


Fig. 9. Hard fault models designed for a transistor Q.

For this circuit, some hard faults can lead to identical results and these hard faults form an ambiguity group (AG). For instance, Q1CEs (or Q2CEs) can result in Q2 shut down and the potential of Vout will be pulled up to Vcc (+12 V). Hence, Q1CEs and Q2CEs are added to the same AG (note that Q4BCs does not exist). For Q1~Q4, the hard fault classes including the AG members are summarized in Table 3. In this experiment, altogether 14 fault classes are considered including the fault-free class (*i.e.* "nf" class).

Table 3. The AGs for the amplifier circuit.

Fault class	AG members
Q1BCs	-
Q1BEs	-
Q1CEs	Q2CEs
Q1Bo	Q1Co,Q1Eo
Q2BCs	-
Q2BEs	Q2Bo,Q2Co,Q2Eo,Q3BEs (Q4BEs), Q3Bo,Q3Co,Q3Eo, Q4CEs
Q3BCs	Q3CEs, Q4Bo,Q4Co,Q4Eo

In order to further evaluate our methods, a simple half-wave rectifier (see Fig. 10) with discrete components is used to obtain real samples and the fault injection is conducted manually. This nonlinear circuit has been studied in [3].

The faulty responses from V_o are measured with a data acquisition card (DAC) under the stimulus of $V_i = \sin(2\pi \cdot 50 \cdot t)$. In our experiment, R1 and R2 produce soft faults and diodes D1, D2 produce hard faults. The fault classes used in this experiment are listed in the following order: nf, R1↓, R1↑, R2↓, R2↑, D1sh, D1o, D2sh and D2o. Here, D1sh means diode D1 shorted and D1o means diode D1 opened. Considering the resistors, the single soft fault model is identical to the first CUT. The faulty samples for the resistors are collected by changing the resistor's value. For each fault class, 50 actual samples are collected.

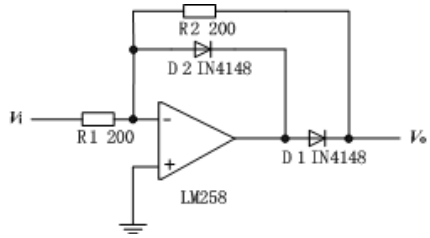


Fig. 10. The actual analog circuit.

4.3 Feature extraction

In our study, wavelet packet decomposition is employed to perform feature extraction and this technique has been addressed in [5] and [8]. For the linear circuits, the WPD technique is applied to the fault samples which are decomposed into approximations and details at level *N* (*N*=1, 2, 3…, segmented with dashed-lines shown in Fig. 11). For this experiment, we have chosen the Haar function as the mother wavelet because this wavelet function works well in our practice.

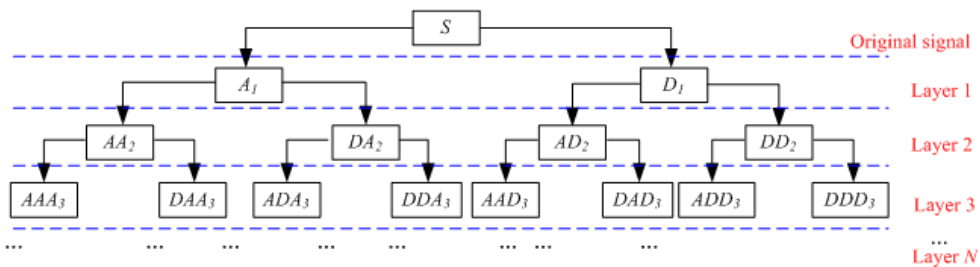


Fig. 11. Wavelet decomposition of a signal *S* into a hierarchical structure.

For the original sample S_{DP}, where DP is the data points of original sample S, for the simulated circuits, DP is 5000 at a sample rate of 5 MHz. For the actual circuit, DP is 2000 when the DAC works at a sample rate of 100 KHz.

In our investigations, the wavelet decomposition is implemented at level five, because, at this depth good results are always achieved. For three simulated circuits, we use the feature extraction approach as follows.

Let the approximation coefficients be $A_i\,(i=1,2,...,16)$, and the detail coefficients be $D_i\,(i=1,2,...,16)$. For every coefficient, the data dimension is $DP/2^5$. We further process these coefficients more compactly:

$$AC_i = \|A_i\|_\infty, \tag{17.1}$$

$$DC_i = \|D_i\|_\infty, \tag{17.2}$$

where $\|\cdot\|_\infty$ is the infinity norm. Our feature sample is a 32-dimensional vector: $[AC_1, AC_2, ..., AC_{16}, DC_1, DC_2, ..., DC_{16}]$.

For the actual nonlinear circuit, the wavelet analysis is used to denoise the collected circuit responses as shown in Fig. 12. Generally, the signals obtained from the DAC are always superimposed with noise. The noise can make fault class (especially for the soft classes) overlap in the measurement space, and this will add difficulties to the subsequent process.

In this study, the wavelet packet is used to decompose the signal into approximations and details. The detail coefficients can be viewed as the high-frequency component of the original signal [6]. In our experiment, the approximations should be retained because they can reconstruct a purified outline of the waveform.

We obtain the first approximation at level five, because at this level, the effect of noise can be negligible. The waveforms reconstructed by the first approximation are shown in Fig. 13, in which we can observe the waveforms have been filtered and refined from noise.

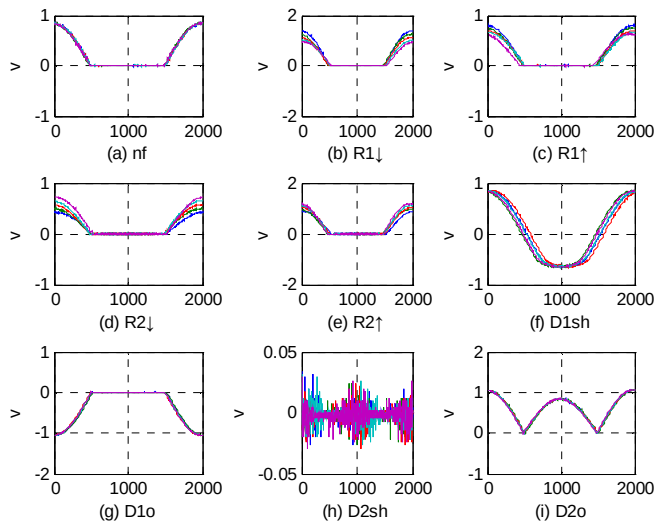


Fig. 12. Waveforms of fault classes for the actual circuit.

Considering that the output waveforms of the circuit are cyclical and simple, we further extract three well-known features in the time domain:

$$[f0, f1, f2]$$

where:

- $f0 = \min(S)$ means getting the minimal value from S;
- $f1 = \max(S)$ means selecting the maximal value from the signal S;
- $f2 = \text{mean}(S)$ is defined as $\left(\sum_{i=1}^{DP} s_i\right) / DP$.

This feature extraction technique is also directly applied to the original samples shown in Fig. 12, in which the waveforms are not preprocessed with wavelet analysis. We did this for the effectiveness illustration of wavelet analysis in terms of noise eradication, because, three features used in this study are easily affected by the noise.

For both cases, we give their three-dimensional (3-D) scatter plots, which are shown in Fig. 14 a and b respectively. In Fig. 14, only five classes (including nf and soft classes) are given because the noise plays a significant effect on the soft classes.

Also in Fig. 14, it is easy to find that the features with the wavelet analysis become more separable and this indicates the effectiveness of the wavelet analysis in terms of noise elimination. Also, this promotion will be beneficial to the subsequent machine learning and pattern classification. This conclusion will be supported by the following results.

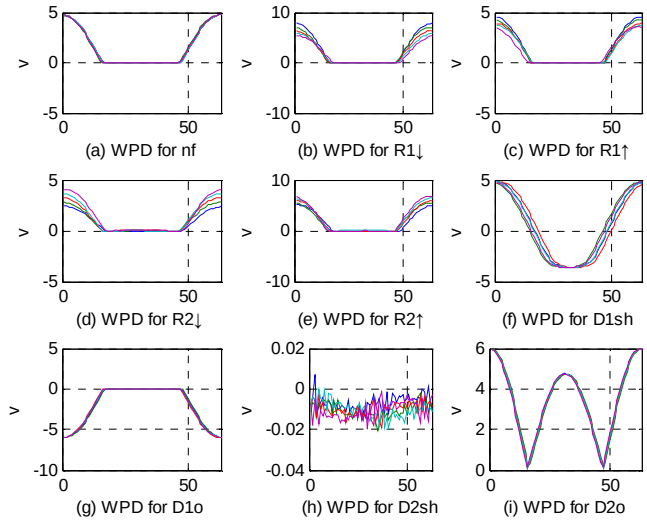
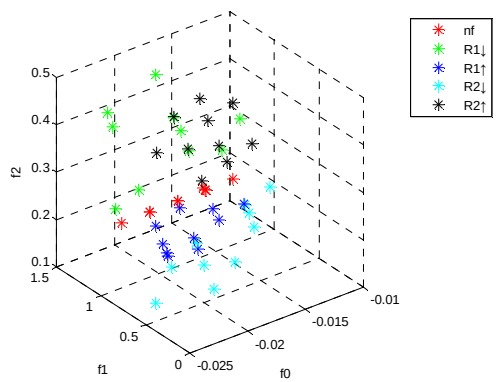


Fig. 13. The first approximation of wavelet analysis at level five for various fault classes.

a)



b)

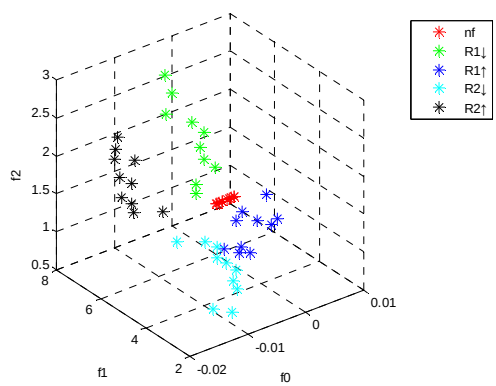


Fig. 14. Feature comparison in 3-D space for two methods: a) features without wavelet analysis; b) features with the wavelet analysis at level five.

5. Results

5.1. Classifier design

In our study, altogether four fault classifiers are designed. The first one is the BPNN with three-layered architecture I-J-K as shown in Fig. 15, in which I is the input size, J is the number of hidden-layer neurons, K is the number of output layer neurons. Also in Fig. 15, b_{hj} is the bias of the j^{th} hidden-layer neuron where $j = 1, 2, ..., J$; b_{ok} is the bias of the k^{th} output-layer neuron whose output is O_k, where $k = 1, 2, ..., K$. $f(\cdot)$ is the activation function of the hidden layer and $g(\cdot)$ is the activation function of the output layer.

Let the input vector be $x = [X_1, X_2, ..., X_I]^T$. The j^{th} hidden layer neuron receives a total activation of $\mathbf{W}_j^T \cdot x + b_{hj}$ from the input layer (i.e. the sample x), where $\mathbf{W}_j = [W_{1j}, ..., W_{Ij}]^T$ is a weight vector connecting the input vector x and the j^{th} hidden layer neuron. The k^{th} output layer neuron receives a total activation of $(\mathbf{W}_k^{'})^T \cdot x^{'} + b_{ok}$, where $x^{'} = [f(\mathbf{W}_1^T \cdot x + b_{h1}), ..., f(\mathbf{W}_J^T \cdot x + b_{hJ})]^T$. Hence, the k^{th} output is:

$$O_k = g((\mathbf{W}_k^{'})^T \cdot x^{'} + b_{ok}). \tag{18}$$

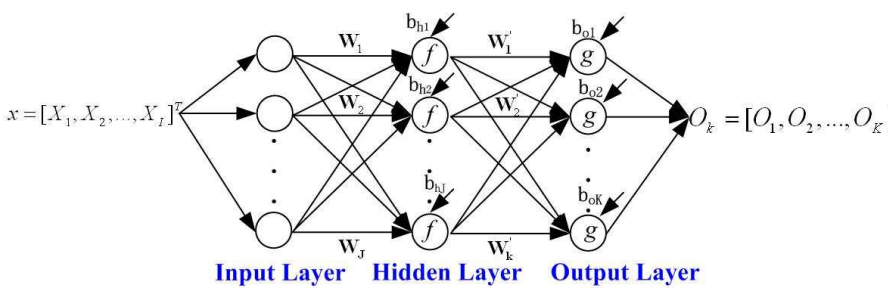


Fig. 15. The BPNN architecture used.

For the target value of the output vector we use a 1-of-*K* coding scheme. Hence, the assignment of x can be determined by:

$$\arg\max_{k=1,2,...,K}(O_k). \tag{19}$$

For the first simulated circuit, we use an architecture of 32-9-9 BPNN to train the samples. For the second simulated circuit and the amplifier circuit, the BPNN has an architecture of 32-13-13 and of 32-14-14, respectively. For the actual circuit, the BPNN uses a structure of 3-9-9.

The first SVC designed in our study is the conventional one-against-rest SVC. The second one is the proposed method in this paper. Two one-against-rest SVCs have identical training parameters, *e.g.*, support vectors, training time, Lagrange multipliers, etc. The third classifier is the one-against-one SVC. In the testing stage of one-against-one SVC, we adopt the decision strategy addressed in [14].

5.2. Machine learning

For all circuits under consideration, the training set contains 10 samples for every fault class. The other remaining 40 samples are used for future testing. All samples are normalized to have zero mean and standard unity variance.

For the BPNN, the activation function tansig is used from the input layer to hidden-layer. From the hidden-layer to output layer, the activation function is logsig. For each BPNN, an error goal of 0.01 is specified because a very small error goal will probably lead to an over-fitted classifier. In our design, a fast training algorithm adjusted by a momentum constant M_c is applied and the BPNN learns well if M_c is chosen within the range of 0.85~0.88. For the BPNN, another problem is that different trainings will probably result in different performances. In this paper, every BPNN is trained three times and the final performance is achieved by selecting the best one from three trainings.

For the SVC, we adopt the standard support vector machines algorithm [18]. In our study, the upper bound *C* is confined to 1000, because with this parameter, the SVC classifier can achieve good classification performance. We mainly investigate the performance of the RBF and polynomial kernel functions. In our study, σ varies across {0.01,0.1,1,2,4,8,16,32, 128,256}and q is confined to {1,2,3}, we select an optimal σ or q depending on the classification performance of the classifier.

5.3 Experimental Results

We write Matlab7.0 codes for all classifiers in terms of sample preprocessing, training and testing, etc. All codes run on a PC with PIV 2.8GHz Duo CPU and 2GB RAM. For the convenience of performance comparison, several specifications are predefined and used in Table 4. Additional explanations are also listed in the remarks column.

In the training stage, for the first two filters, the BPNN converges to the error goal, but, for the other two circuits, the BPNN cannot converge within the specified epochs. Despite this, the BPNN can still be used to perform fault classification. In our study, for all the circuits under test, the SVC can converge to a global solution quickly. The training time comparisons for these classifiers are also given in Table 5.

Table 4. Specifications used in our experiments.

Specifications	Definitions	Remarks
Accuracy	percentage of correct classification only for testing samples	
Recall	Percentage of correct classification only for training samples	
TeT	testing time	in seconds
TrT	training time	in seconds
NSF	the number of samples falling into the URs or the RR	only used for the SVCs
TNB	The total number of BSVCs used for testing certain fault class	only used for the 1st SVC and the 2nd SVC and 40 samples are tested.

Our comparisons for different classifiers are mainly based on the specifications.

– **Accuracy & Recall & TeT & TrT**. Focusing on the specifications, we give the results by BPNN the SVCs in Table 5. In testing the Sallen-Key and High-Pass filter, we found that the first-order polynomial kernel function (q=1) can give the best performance. For the

differential amplifier, we choose $\sigma = 16$. For the actual rectifier circuit without wavelet analysis, σ=0.1 and for the same circuit by employing wavelet analysis to get rid of noise, the 3rd-order polynomial kernel function is the best choice.

Table 5. Experimental results of several classifiers for the circuits.

Circuit	Classifier	Accuracy	TrT	TeT	Recall
The Sallen-Key filter	BPNN	0.972	2.135	0.329	1
	The 1st SVC	0.978	1.762	6.412	1
	The 2nd SVC	0.983	1.762	2.271	1
	The 3rd SVC	0.992	0.879	18.864	1
The High-Pass filter	BPNN	0.998	3.210	0.443	1
	The 1st SVC	1	1.268	18.023	1
	The 2nd SVC	1	1.268	2.932	1
	The 3rd SVC	1	1.374	56.227	1
The differential amplifier	BPNN	0.964	15.228	0.646	0.979
	The 1st SVC	0.963	1.956	19.894	1
	The 2nd SVC	0.964	1.956	4.332	1
	The 3rd SVC	0.966	2.388	74.378	1
The Half-wave-rectifier (without wavelet analysis)	BPNN	0.836	14.4	0.267	0.856
	The 1st SVC	0.828	1.147	4.250	1
	The 2nd SVC	0.828	1.147	0.787	1
	The 3rd SVC	0.847	0.894	9.340	1
The Half-wave-rectifier (wavelet analysis)	BPNN	0.936	11.624	0.263	0.956
	The 1st SVC	0.969	1.762	4.335	1
	The 2nd SVC	0.978	1.762	0.820	1
	The 3rd SVC	0.992	0.879	9.438	1

In Table 5, for most cases, the SVC classifier achieves a very close performance (*i.e.* accuracy) to the BPNN with wavelet analysis as the preprocessor. For the actual circuit with wavelet analysis, the SVC displays an excellent and accurate the classification performance, which is apparently superior to the BPNN. Also, the SVC gives 100% recall capability compared to the BPNN. These data indicate that the SVC can be applicable to the analog fault detection and localization as an alternative for the BPNN. In our investigations, some samples fail to be classified because their corresponding output waveforms are almost identical.

For the SVCs, the one-against-one SVC always gives an excellent generalization capability, however, this classifier consumes too much time to perform classification task. Compared to the conventional one-against-rest and one-against-one SVC, the proposed approach can give a comparable performance but it needs far less time to implement testing.

In addition, the SVC illustrates different performances depending on different feature extractors when it goes to the actual circuit. For the features without wavelet analysis, the classifier displays an inferior performance. However, the classifier performs quite well when it is applied to the features with wavelet analysis at level five. This difference also validates the effectiveness of the wavelet analysis in terms of noise eradication.

– **TNB.** This specification is mainly used to evaluate the computational complexity of two one-against-rest SVC. For the conventional one-against-rest SVC, forty testing samples will result in the anticipation of total $N \cdot 40$ BSVCs, where N is the number of fault classes. For the proposed method, only a small portion of BSVCs are needed. Hence, based on the aforementioned analysis of the presented flow chart, the time needed is reduced remarkably and this reduction is also clear in Table 5. The TNBs consumed by two methods are shown in Fig. 16, in which the discrepancy between the first SVC and the second SVC seems to be distinct.

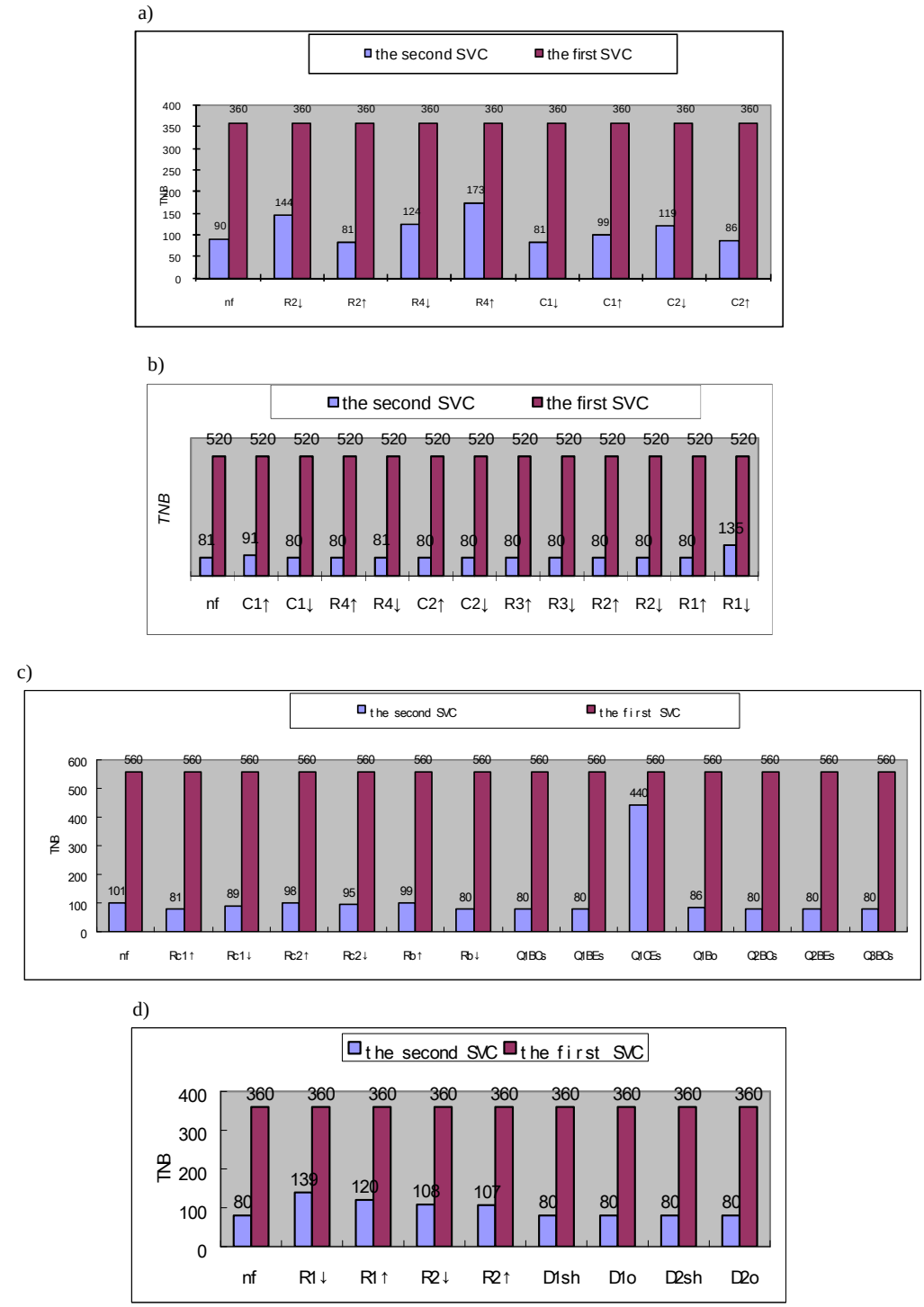


Fig. 16. Comparison of *TNB*s consumed by the 1st SVC and the 2nd SVC in testing a) the Sallen-Key circuit (*N*=9); b) the High-Pass filter (*N*=13); c) the differential amplifier (*N*=14) and d) the actual nonlinear circuit with wavelet analysis (*N*=9).

– **NSF.** This specification is mainly used to detect the samples falling into the RR. A large NSF always means a larger computational complexity. In our investigations, the NSF seems to be small. In our method, the space distance based approach is employed to resolve these problems. From Table 6, it is clear that the proposed fault decision algorithm can correctly classify most of the samples falling into the URs or the RR. In diagnosing the actual circuit, the NSF is reduced to 13 from 17 after wavelet analysis, because the wavelet analysis can eliminate the effect of the noise, and this elimination can make the features become distinguishable and thus, fewer samples fall into the URs or the RR.

Table 6. The effectiveness of the space distances decision algorithm.

Circuit	*Number of testing samples*	*NSF*	*Number of samples correctly classified using the decision algorithm*
The Sallen-Key filter	360	21	19
The high-pass filter	520	7	7
The differential amplifier	560	36	32
The Half-wave-rectifier (without wavelet analysis)	360	17	12
The Half-wave-rectifier (with wavelet analysis)	360	13	8

6. Conclusions

In this paper, we investigate the diagnosis performance of the SVCs by using fault dictionary methods. Useful conclusions can be drawn by reviewing the above results:

– The SVC can be used to perform an analog circuit diagnosis task. In our research, the important parameters for the SVC are mainly upper bound C and kernel function parameters. The support vectors, as well as the corresponding Lagrange multipliers can be found automatically by the training algorithm. For the BPNN, too many network parameters need to be adjusted manually (*e.g.*, the hidden layer neurons, activation function types, learning rate, momentum constant, *etc*.), thus, resulting in a more unreliable classifier structure. In addition, in our investigations, the trainings of SVCs are always successful and the training time needed for each training set is also stable, but the BPNN sometimes fails to converge and its performance also varies depending on the training stage. In addition to this, the SVC illustrates an excellent and stable fault classification performance, which is close or even superior to the BPNN.

– The wavelet packet analysis is useful in our investigations and this usefulness lies in that it can effectively perform feature size reduction and noise eradication operations. In our study, the wavelet mother function is Haar, and wavelet analysis depth is specified at level five, because with this, good results are always achieved. These results also indicate that the Haar wavelet function is effective in the application of analog circuit faults classification. In further research, other types of wavelet functions will be exploited.

– The simulated and practical results have shown that the SVCs, including our proposed methods, are applicable to analog circuit diagnosis. The one-against-one SVC performs well in our diagnosis cases, but it is not suitable for analog diagnosis case with a large number of fault classes, because this classifier requires $N(N-1)/2$ BSVCs. If N is very large, this approach will become prohibitive. Our proposed method is based on the one-against-rest SVC, which requires N BSVCs to determine a sample assignment. Generally, according to the above results and analysis, the proposed method needs less computational cost to perform a fault classification task.

– The computational complexity of the SVC depends on the number of support vectors, thus leading to different testing time. This can be roughly explained through Eq. (16).

Generally speaking, a large number of support vectors will result in high computational and storage complexities. Hence, reducing the number of support vectors while maintaining the classifier performance unchanged seems to be a prominent task and this task will be envisaged in our next work.

Acknowledgments

This work was supported by the National Nature Science Foundation of China (grant # 60871009), by the Aviation Science Foundation of China (grant # 2009ZD52045), and by NUAA Research Funding of China (grant # NS2010063).

References

[1] Catelani, M., Fort, A., Alippi, C. (2002). A fuzzy approach for soft fault detection in analog circuits. *Meas*., 32, 73-83.

[2] Spina, R., Upadhyaya, S. (1997). Linear circuit fault diagnosis using neuro-morphic analyzers. *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, 44(3),188-196.

[3] Aminian, F., Aminian, M. (2001). Fault Diagnosis of Nonlinear Analog Circuits Using Neural Networks with Wavelet and Fourier Transforms as Preprocessors. *J. Electron. Test.: Theory Appl.*, 17, 471-481.

[4] El-Gamal, M.A., Abu El-Yazeed, M.F. (1999). A Combined Clustering and Neural Network Approach for Analog Multiple Hard Fault Classification. *J. Electron. Test.: Theory Appl.*, 14, 207–217.

[5] Yanghong, T., Yigang, H. (2008). A novel method for fault diagnosis of analog circuits based on WP and GPNN. *Int. J. Electron.*, 95(5), 431-439.

[6] Aminian, M., Aminian, F. (2007). A Modular Fault-Diagnostic System for Analog Electronic Circuits Using Neural Networks With Wavelet Transform as a Preprocessor. *IEEE Trans. Instrum. Meas.*, 56(5), Oct., 1546-1554.

[7] Fanni, A., Giua, A., Marchesi, M., Montisci, A. (1999). A Neural Network diagnosis Approach for analog circuits.*Appl. Intell.*, 11(2), 169-186.

[8] Yigang, H., Yanghong, T., Yichuang, S. (2004). Fault Diagnosis of Analog Circuits based on Wavelet Packets. In *Proceedings of IEEE TENCON*. Thailand, 267-270

[9] Catelani, M., Fort, A. (2002). Soft Fault Detection and Isolation in Analog Circuits: Some Results and a Comparison between a Fuzzy Approach and Radial Basis Function Networks. *IEEE Trans. Instrum. Meas.*, 51, (2), 196-202.

[10] Salat, R., Osowski, S. (2003). Analog Filter Diagnosis Using Support Vector Machine. In *Proceedings of ECCTD*. Krakow, Poland, 421-424.

[11] Siwek, K., Osowski, S., Markiewicz, T. (2006). Support Vector Machine for Fault Diagnosis in Electrical Circuits. In *Proceedings of NORSIG*. Iceland, 342-345.

[12] Grzechca, D., Rutkowski, J. (2009). Fault Diagnosis in Analog Electronic Circuits – The SVM Approach. *Metrol. Meas. Syst.*, 16(4), 583-598.

[13] Vapnik, V.N. (1998). *Statistical Learning Theory*. New York: Wiley.

[14] Hsu, C.W., Lin, C.J. (2002). A Comparison of Methods for Multi-class Support Vector Machines. *IEEE Trans. Neural Networks*, 13(2), 415-425.

[15] Chapelle, O., Haffner, P., Vapnik, V.N. (1999). Support Vector Machines for histogram-based image classification. *IEEE Trans. Neural Networks*, 10(5), 1055-1064.

[16] Takahashi, F., Abe, S. (2002). Decision-Tree-Based Multi-Class Support Vector Machines. In *Proceedings ICONIP*. Singapore, 1418-1422.

[17] Burges, C.J.C. (1998). A Tutorial on Support Vector Machines For Pattern Recognition. *Data Min. Knowl. Disc.*, 2(2), 121-167.

[18] *http://www.princeton.edu/~kung/ele571/571-MatLab/571svm/*[as of Dec. 2008].