Xiao *et al. EURASIP Journal on Image and Video Processing* (2017) 2017:48
DOI 10.1186/s13640-017-0197-y

EURASIP Journal on Image and Video Processing

RESEARCH Open Access

An evolutionary classifier for steel surface defects with small sample set

Mang Xiao[1], Mingming Jiang[2*], Guangyao Li[2], Li Xie[2] and Li Yi[2]

Abstract

Nowadays, surface defect detection systems for steel strip have replaced traditional artificial inspection systems, and automatic defect detection systems offer good performance when the sample set is large and the model is stable. However, the trained model does work well when a new production line is initiated with different equipment, processes, or detection devices. These variables make just tiny changes to the real-world model but have a significant impact on the classification result. To overcome these problems, we propose an evolutionary classifier with a Bayes kernel (BYEC) that can be adjusted with a small sample set to better adapt the model for a new production line. First, abundant features were introduced to cover detailed information about the defects. Second, we constructed a series of support vector machines (SVMs) with a random subspace of the features. Then, a Bayes classifier was trained as an evolutionary kernel fused with the results from the sub-SVM to form an integrated classifier. Finally, we proposed a method to adjust the Bayes evolutionary kernel with a small sample set. We compared the performance of this method to various algorithms; experimental results demonstrate that the proposed method can be adjusted with a small sample set to fit the changed model. Experimental evaluations were conducted to demonstrate the robustness, low requirement for samples, and adaptiveness of the proposed method.

Keywords: Surface defect, Support vector machine, Random subspace, Bayes classifier

1 Introduction

With the growth in competition among producers of steel strip, the quality of the steel strip surface has become very important. Steel strip quality and the surface quality of structural products have assumed increasingly significant importance [1]. The importance of surface quality requires that effective and efficient methods be implemented to replace conventional artificial visual inspection during which an expert can only inspect 0.05% of the total steel surface, which can be easily impacted by fatigue and other unfavorable conditions. Artificial inspection cannot satisfy the quality requirements. Therefore, automatic, high-accuracy steel surface inspection systems have become essential to the production system.

Surface defect inspection systems mainly consist of two parts: defect segmentation and defect processing. Defect processing entails feature extraction and defect classification. In recent years, abundant research has been conducted, and different kinds of feature extraction and classification methods have been introduced to classify steel strip surface defect. In [2, 3], the defects were classified by using K-nearest neighbor (KNN) methods with a co-occurrence matrix. Santanu Ghorai et al. [4] described an automated visual inspection system with discrete wavelet transform (DWT) features and a support vector machine (SVM). Wu et al. [5] described an algorithm with an undecimated wavelet transform (UWT) and a mathematical morphology to detect geometric defects that achieved a 90.23% accuracy that is difficult to achieve in real industrial application in 2008. However, in 2013, a noise-robust method based on a completed local binary proposed by Song [6] averaged 98% accuracy and was shown to be effective enough to be applied to real production. A review of vision-based steel surface inspection systems [1] indicates that most such systems have achieved >90% accuracy.

For application to real production, some difficulties for steel surface defect detection remain. In a real production environment, different products can be produced on a single production line, and some products are manufactured

*Correspondence: stupidhod@hotmail.com
[2]Department of Electronics and Information Engineering, Tongji University, Shanghai, China
Full list of author information is available at the end of the article

on several production lines at the same time. With the passage of time, the physical conditions of the equipment and of the detection device will both change. Each of these variables has only a small impact on the real-world model; however, the classifier trained by the original database will not work well on the changed real-world model. If we trained every production line and every product with a single database, it would take a long time for a steel company to get a large enough sample set for a newly built production line without a trained inspection system.

Unfortunately, most research has focused on algorithms that work only when a large number of samples is available, there little research has focused on this production problem. Solly et al. proposed a rapidly evolving system with an expert's feedback; however, that research describes an adaptive interactive evolution methodology for determining parameters to control segmentation of surface defects on images, and the classification accuracy will still be impacted by changing the real-world model.

Therefore, to solve these issues, we propose an evolutionary classifier with a Bayes kernel (BYEC) that can be adjusted with small sample sets to fit the changed model. Because the small changes to the real-world model will impact some features of the classifier, the misclassification of the classifier is mainly impacted by these features. Nevertheless, if we reduce the impact of these features, we should be able to restore the classifier to get a relatively high accuracy [7]. Thus, we built a classic classifier with an evolutionary Bayes kernel and adjusted the kernel with a small sample set for the changed production model rather than training every product and every production line with a single data set.

Our technical contribution includes three points. First, we propose an evolutionary classifier with a Bayes kernel (BYEC) for steel defect classification. Second, we adopt multiple SVMs to predict individual features, enabling our evolutionary Bayes kernel to fit well with a small sample set for the changed production model. Third, and most importantly, we combine five selected features and prove that our method has high accuracy, is adaptive, and has a low requirement for samples in our experiments.

The rest of the paper is organized as follows: Section 2 depicts the procedures of the surface defect inspection system. Section 3 presents the features we used in this paper. The method to build SVM subclassifiers on a random subspace of features is provided in Section 4. The method to build the Bayes kernel and evolve the kernel is discussed in Section 5. Section 6 gives comparative experimental results of this algorithm and research on factors that affect the results. Finally, we conclude this paper in Section 7 and give some suggestions for future work.

2 Structure of the evolutionary classifier for steel surface defects

An evolutionary surface defect inspection system should be trained as a classic classifier and evolve to fit the changed real world on the real production line with a small sample set. The structure of the inspection system is presented in Fig. 1. Our system mechanism mainly involves three processes: defect acquisition, feature extraction, and defect classification.

The image acquisition system has been discussed in much of the literature [8–10] and has become a mature field, so we will not discuss it in this paper again.

As it illustrated in the Fig. 1, five kinds of features were introduced in this system. The reason of this is because the penalize of some features in the adjustment process drop some useful information of the defect and may lead to misclassification, so superfluous features were imported. We integrated the uniform local binary patterns (ULBP) [11], gray-level co-occurrence matrix (GLCM) features [12], the histogram of oriented gradient (HOG) feature, and gray-level histogram and Gabor filter features. These abundant features guarantee enough information for the classifier. To utilize these features, a multiple classifier system was imported into this classifier.

The classification part has two components in Fig. 1: multiple SVM classifiers and a Bayes kernel classifier. Compared with the classic steel surface inspection system, the fuser Bayes kernel makes the key contribution to this system by fusing the results from the multiple classifiers and adjusting the hybrid parameters.

The SVM [13] is a popular small sample set learning method. It offers very good performance for pattern classification problems by minimizing the Vapnik-Chervonenkis (VC) dimension and achieving a minimal structural risk. Because of its small sample set requirement, fast learning capability, and good performance, SVM is a good choice for this method. Because each SVM classifier is trained by a subspace of the feature space, we can evaluate and penalize the features by using the corresponding SVM classifier.

A multiple classifier system (MCS) offers many alternatives for unorthodox handing of realistic complex problems [14], allowing us to exploit the potential of the individual classifiers and get enhanced performance by their combination. It can perform better than the individual classifiers and is easily implemented on parallel, multithreaded, and distributed architectures, which is very important for the real-time production environment. The critical aspect of a MCS is that it is well suited to treating drift, which means that the statistical dependencies between object features and its classification may change

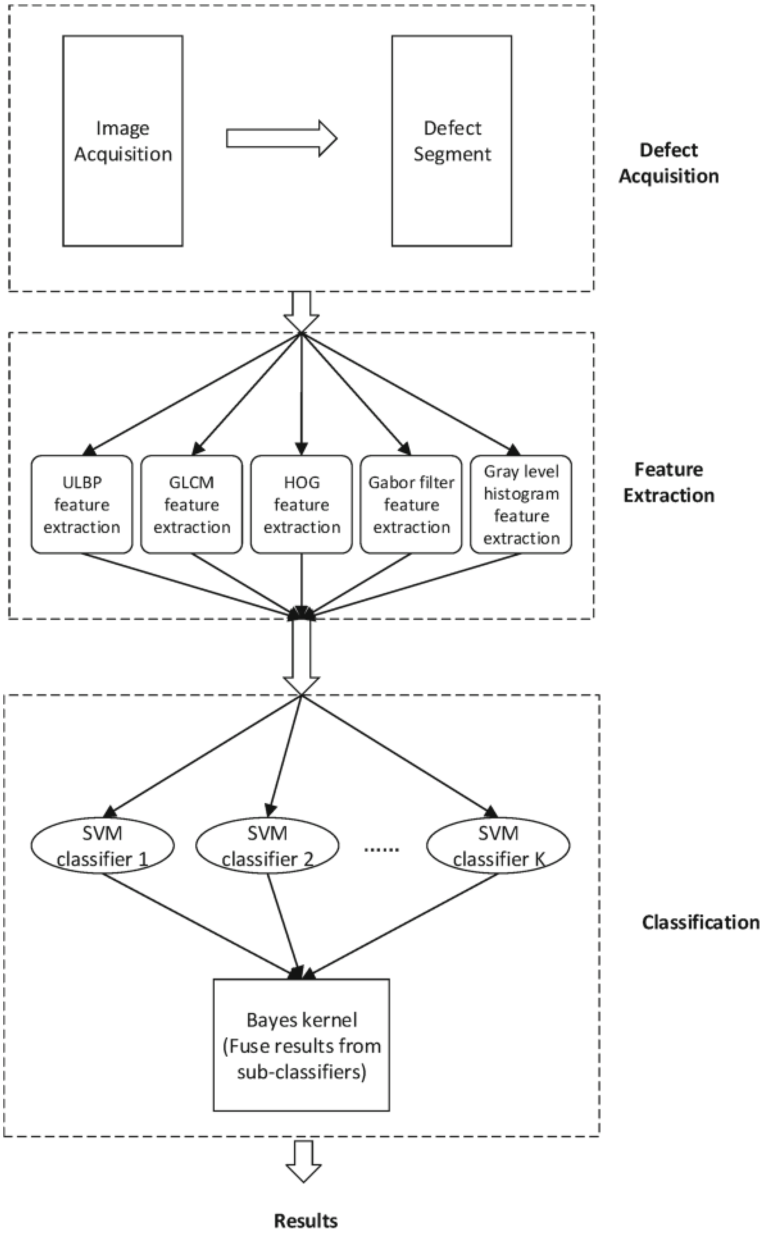


Fig. 1 Structure of the evolutionary classifier for steel surface defects using a Bayes kernel

in time so that future data may be badly processed if we maintain the same classification. Drift decreases the accuracy of the classification result, the individual classifier evaluation is done on their accuracy on the new data. The best performing classifiers are selected to constitute the MCS committee after every loop. Kolter et al. [15] described a dynamic weighted majority algorithm. We also use an evolvable weighted method to change our classifier with time.

The method to fuse the subclassifiers must be adjusted with a small sample set and the accuracy of the subclassifiers must be evaluated; therefore, the Bayes classifier is a good candidate as it builds a reference from prior probability to posterior probability. We can score the performance of the classifiers on the changed model by the posterior probability. With a small sample set, we can get the approximate posterior of the classifiers and use it to reweight the integrated classifier to fit the real model. A Bayes kernel was built based on this concept.

Figure 2 depicts the processes to adjust the Bayes kernel, which is executed in three steps. First, defect images are classified by the SVM subclassifiers, then the defect images are classified by the expert, and, finally, the results of the classifiers and the corresponding tagged data are

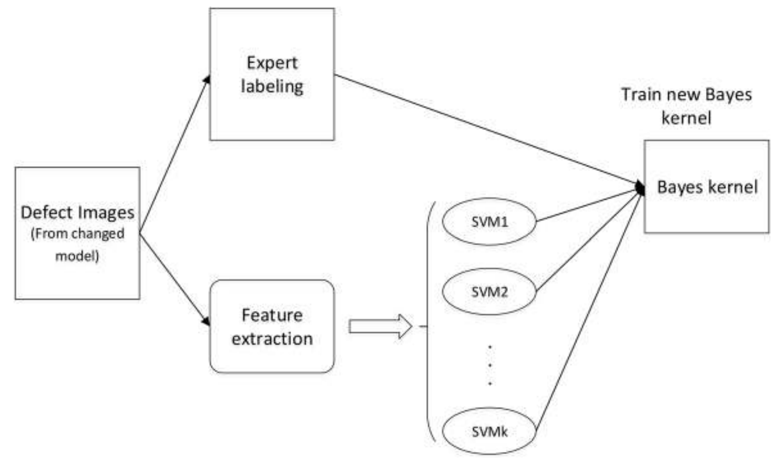


Fig. 2 Process to adjust the classifier with samples from a changed production line

used to train the Bayes kernel classifier. This Bayes classifier is combined with the multiple SVM classifiers as an integrated classifier.

3 Extraction of features

To avoid the lack of information after integration, we introduce redundant features to overcome this weakness. Five different kinds of features are extracted in the inspection system to describe the properties of texture, color, and shape, respectively. The feature space consists of a gray-level co-occurrence matrix, a uniform local binary pattern, a histogram of oriented gradient, a gray-level histogram, and a Gabor filter.

3.1 Gray-level co-occurrence matrix

Gray-level co-occurrence matrix is a $N_g \times N_g$ matrix where N_g is the number of gray levels in the image, this matrix reflects the direction, adjacent distance, and change range of the image. Every value in the matrix represents a joint probability that two gray-level pixels exist with a distance of d and a direction θ, it is defined in Eq. 1. Where matrix C is computed over an $n \times m$ image I , d and θ are described by Δx and Δy that $\Delta x = sin(\theta) \times d$, $\Delta y = cos(\theta) \times d$. Where i and j are the image intensity values of the image, p and q are the spatial positions in the image.

$$C_{\Delta x,\Delta y}(i,j) = \sum_{n}^{p=1}\sum_{m}^{q=1} \begin{cases} 1, \text{ if } I(p,q) = i \text{ and } I(p+\Delta x, q+\Delta y) = j \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{1}$$

The GLCM is sensitive to rotation, so we choose four directions 0°, 45°, 90°, and 135° to cover more information and select a distance of 8. The dimensions of the GLCM is too large for us to process, so we choose Haralick features [12] to describe it which calculates the correlation, energy, contrast, entropy, and inertia quadrature of the GLCM. Finally, we get a vector to describe the GLCM feature.

3.2 Histogram of oriented gradient and gray level

The histogram is a key tool in image processing, it is one of the most useful techniques in gathering information about a matrix. The gray-level histogram of the defect image represents the distribution of the pixels over the gray-level scale and reflects the contrast, gray level, and other information of the image. It can be visualized as if each pixel is placed in a bin corresponding to the color intensity of that pixels. We make the histogram of the defect image to 40 bins and turn it into a feature vector as the gray-level feature.

Dalal Navneet and Triggs Bill proposed the histogram of oriented gradient in 2005 [16], this feature is used in computer vision and image processing for object detection, which counts the occurrences of the oriented gradient in an image. The image can be described by the distribution of intensity gradients and edge directions. In conventionally procedure, the image is divided into many small cells and calculated respectively, but in this paper, we make the histogram on the whole picture.

$$\begin{aligned} G_x &= H(x+1,y) - H(x-1,y) \\ G_y &= H(x,y+1) - H(x,y-1) \\ G(x,y) &= \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \\ \alpha(x,y) &= \tan^{-1}\left(\frac{G_y(x,y)}{G_x(x,y)}\right) \end{aligned} \tag{2}$$

Equation 2 defines the gradient orientation and value, G_x is the gradient value of x orientation, G_y is the gradient value of y orientation, $G(x,y)$ represents the gradient value, and $\alpha(x,y)$ represents the orientation of the pixel. As the Fig. 3 described, we firstly convert the defect image to the orientation image (gray level range from 0° to 180°),

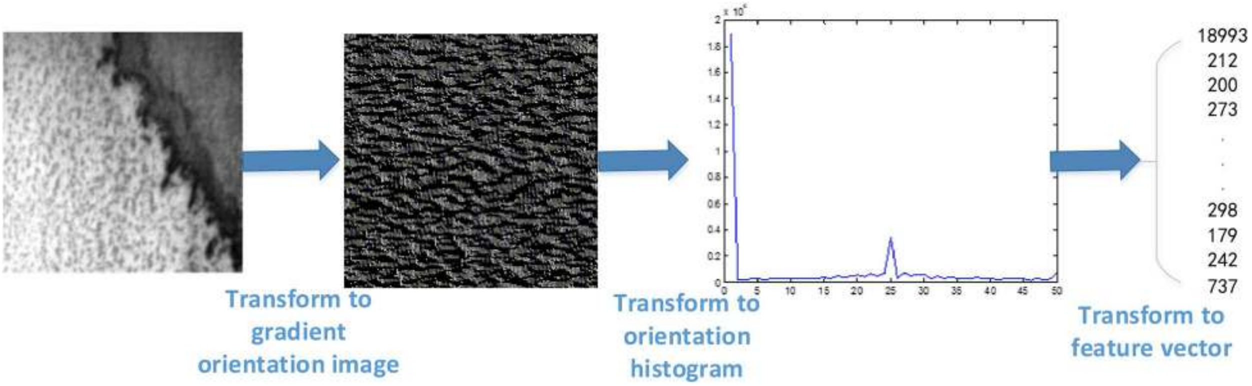

Transform to gradient orientation image
Transform to orientation histogram
Transform to feature vector
x 10⁴
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
25
30
35
40
45
50
18993
212
200
273
.
.
298
179
242
737

Fig. 3 Process to convert defect image to HOG feature vector

then we make the histogram of the image with 50 channels. Finally, we get the value of every bin and convert it to a feature vector to represent the HOG.

3.3 Uniform local binary pattern

The local binary pattern (LBP) is one of the most successful statistical approaches for texture classification due to its gray-scale and rotation invariance, this feature reflects the local texture of the image. LBP filter is a 3 × 3 window. As it defined in Eq. 3, the gray level of the center pixel is set as the threshold, gray value of the adjacent 8 pixels around center pixel is compared with the threshold. If the value of a surrounding pixel is larger than the threshold, the position of this pixel is marked as 1, otherwise 0.

$$\mathrm{LBP}_P = \sum_{p=0}^{P-1} s(g_p - g_c)2^p, s(x) = \begin{cases} 1 & \text{if } x \geqslant 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{3}$$

One local binary filter can produce 2^p different values, and this dimension is too high for us to process. Ojala [11] proposed ULBP to reduce the dimension of the LBP. ULBP introduces a uniformity measure U which corresponds to the number of spatial transitions (between [0, 1]) in the pattern. For example, $U(00000001_2)$ and $U(00000010_2)$ equal 2 as they have two transitions between [0, 1] . The U value of most LBPs is not greater than 2, and the number of these LBPs is 57. The ULBP is defined as Eq. 4, a unique id is assigned to a pattern that its U value is not greater than 2, so we reduce the dimension from 256 to 58. As Fig. 4 described, to get the ULBP feature, firstly, we convert the defect image to a ULBP image and then we make the histogram of the ULBP image and convert the histogram to a feature vector.

$$\mathrm{ULBP}_P = \begin{cases} \mathrm{id(LBP)} & \text{if } U\ (\mathrm{LBP}(x)) \geqslant 2 \\ 58 & \text{if } U\ (\mathrm{LBP}(x)) < 2 \end{cases} \tag{4}$$

3.4 Gabor filter

The Gabor filter is used for edge detection in image processing, the frequency and orientation representations of Gabor filters are similar to those of the human visual system and efficient to describe the texture. Its impulse response is defined by a sinusoidal wave multiplied by a Gaussian function. Because of the multiplication convolution property (Convolution theorem), the Fourier transform of a Gabor filter's impulse response is the convolution of the Fourier transform of the harmonic function and the Fourier transform of the Gaussian function. The filter has a real and imaginary component representing orthogonal directions [17]. We only take the real part of the Gabor filter in this paper. The real Gabor filter kernel is defined as Eq. 5.

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \exp\left(-\frac{x'^2 + \lambda^2 y'^2}{2\theta^2}\right)\cos\left(2\pi\frac{x'}{\lambda} + \psi\right) \tag{5}$$

λ is the wavelength of the sinusoidal factor, θ represents the orientation of the normal to the parallel stripes of the Gabor function, ψ represents the phase offset of the sinusoidal function, σ represents the sigma deviation of the Gaussian envelope, and γ represents the spatial ratio. As depicted in Fig. 5, in this paper, we make a Gabor filter bank with $\lambda(2, 3, 4, 5, 6)$ and θ $\left(0, \frac{1}{8}\pi, \frac{2}{8}\pi, \frac{3}{8}\pi, \frac{4}{8}\pi, \frac{5}{8}\pi, \frac{6}{8}\pi, \frac{7}{8}\pi\right)$, totally with 40 Gabor filter kernels, and apply convolution on the defect image with these Gabor filters and get 40 Gabor images. We make a Gabor feature vector composed of the energy of these Gabor images.

4 The SVM subclassifiers on a random subspace of features

After feature extraction, we get a feature vector of dimensions. In general, when the feature dimensions are too large compared with the scale of the sample set, overfitting problems can arise. Therefore, instead of training one classifier to cover all the feature space, we separate the features with a random sampling scheme without replacement and keep almost equal dimensions for every

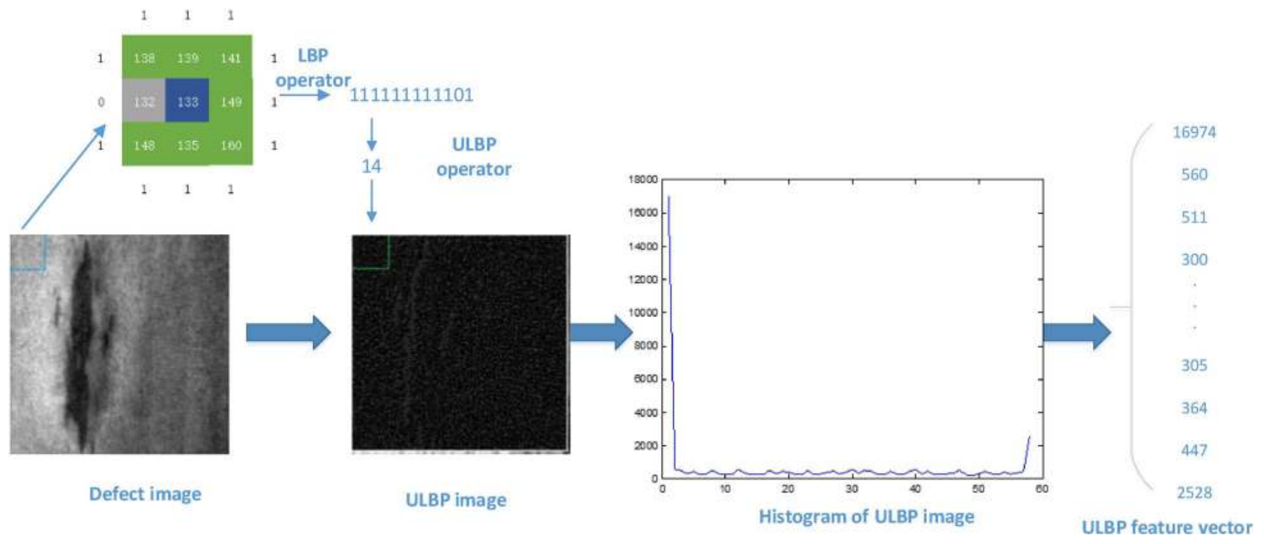


Fig. 4 Process to convert defect image to ULBP feature vector

subspace. We choose a feature subspace with the random sampling scheme rather than with a feature category or in sequence for two reasons: (1) the feature punished will involve features nearby that may contain some useful information and not interfered by the real-world model, and (2) features nearby may contain the same information and the classifier will not get a good result with nearby features.

To overcome the small sample set issue on the production line, we introduce a support vector machine, which has been widely used in many areas, such as computer vision, natural language processing, and neuroimaging, for its good performance, fast training capability, and small sample set requirements for labeled samples.

There are six kinds of defects in the database which we used for the experiment, as SVM is a binary classifier, we implemented the multiclass classification with the “one-against-one” scheme which is usually applied for binary classifier [18, 19]. For every multiclass SVM classifier in this paper, the number of binary SVM classifiers we need

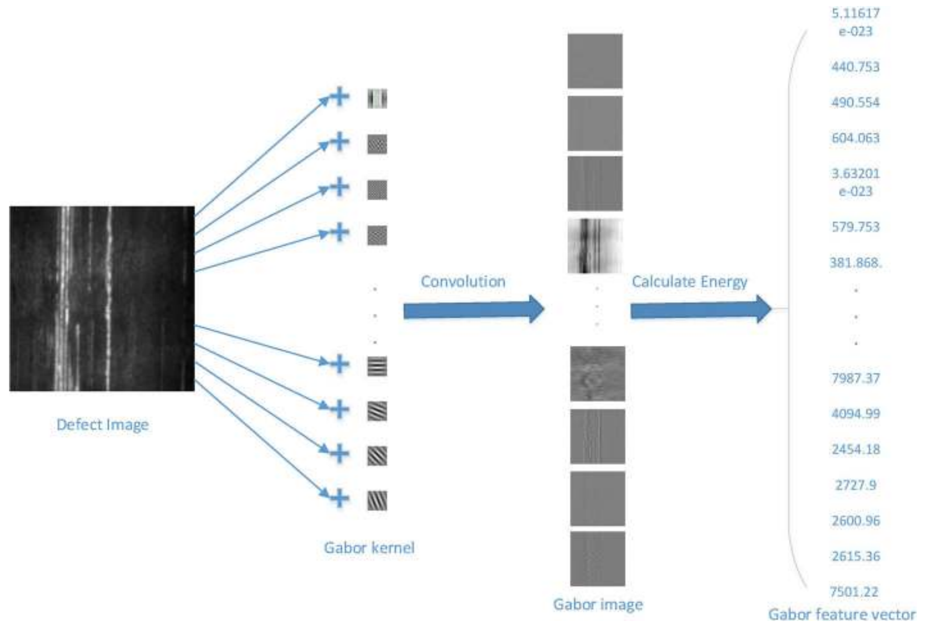


Fig. 5 Process to convert defect image to Gabor feature vector

is defined as Eq. 6, the k is the number of classes exists in data set.

$$n_{\text{SVM}} = k(k-1)/2\,, \text{where } k \geq 1. \tag{6}$$

Corinna describes the standard SVM for two classifications in [20] within the structural risk minimization. The key of SVM is to find the hyperplane to minimize the distance between the two classes to be separated.

With a training vectors $x_i \subseteq R^n, i = 1, \ldots, l$, belong to two classes, and vector $y \subseteq R_l$ such that $y_i \subseteq \{1, -1\}$ indicate the class of the corresponding data, the SVM try to solve the following minimization problem in Eq. 7:

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ &\text{subject to } y_i \left(w^T \phi(x_i) + b \right) \geq 1 - \xi_i \\ &\qquad \xi_i \geq 0, i = 1, \ldots, l, \end{aligned} \tag{7}$$

the ξ_i is a map function which maps x_i into higher feature space make the points easily to be separated and $C > 0$ is the regularization parameter, w is the weight vector for the feature space. To solve the possible high dimensionality of the vector w, we usually convert it to the dual problem as Eq. 8 presents:

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \\ &\text{subject to } y^T \alpha = 0, \\ &\qquad 0 \leq \alpha_i \leq C, \;\; i = 1, \ldots, l. \end{aligned} \tag{8}$$

In which $e = [1, \ldots, 1]^T$ is the vector of all ones, Q represents an l by l positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is the kernel function.

Then, the problem is solved with the primal dual relationship, we can calculate the w with Eq. 9.

$$w = \sum_{i=1}^{l} y_i \alpha_i \phi(x_i) \tag{9}$$

Finally, we can classify the data points with the Eq. 10:

$$\operatorname{sgn}\left(w^T \phi(x) + b \right) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b \right). \tag{10}$$

For the classification of ith and jth classes, we solve it with the Eq. 11 deduced from Eq. 7.

$$\begin{aligned} &\min_{w^{ij}, b^{ij}, \xi^{ij}} \frac{1}{2} {w^{ij}}^T w^{ij} + C \sum_t \left(\xi^{ij} \right)_t \\ &\text{subject to } {w^{ij}}^T \phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}, \text{if } x_t \text{ in the } i\text{th class} \\ &\qquad {w^{ij}}^T \phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}, \text{if } x_t \text{ in the } j\text{th class} \\ &\qquad \xi_t^{ij} \geq 0 \end{aligned} \tag{11}$$

We combine the results from the binary classifier with a voting scheme: every binary classifier has a vote, and a data point is classified into the class with the maximum number of votes. To solve the clash of the same votes, we simply choose the class with the greater sequence number. In addition, we increase the sequence number for the classes with every multiclass SVM to avoid accumulated deviations. A simple example indicates the risks without this strategy: Assume we have three classes, A, B, and C, all with the same accuracy and scale. Then, samples classified into A are least.

5 Combining subclassifiers by using the Bayes kernel

The combination of the subclassifiers is very important to this evolutionary classifier because it not only response for improving the performance of the final integrated classifier but also for the ability to evolve itself to fit the new changed model. Many ensemble methods have been presented [21–23]; however, these methods do not suit this adaptive inspection system. This is because (1) these methods are sensitive to the size of the training sample set (although, even for a mature production line, there are not too many labeled samples), and (2) the fusion of classifiers may be biased from the combination of samples from the changed model.

For these reasons, we propose a new fusion strategy based on a native Bayes classifier. This is a highly practical Bayesian learning method deduced from the Bayesian theorem

$$P(B_i|A) = \frac{P(B_i)P(A|B_i)}{\sum_{i=1}^{n} P(B_i)P(A|B_i)} \tag{12}$$

This theorem was proposed for about 300 years by Thomas Bayes 12 and has developed into a great branch of machine learning. In some domains, it is presented as comparable to neural networks and to other machine learning methods. The naive Bayes classifier $f(x)$ is described by a conjunction of attribute values when the $f(x)$ is limited to a finite set V.

In Bayes learning, the training examples are described by a feature vector $(a_1, a_2, a_3 \ldots a_n)^T$; the Bayes classifier makes decisions based on the probability for every possible value and selects the most portable target.

In this model, we assign every subclassifier as a feature in the feature vector and described it as a probability matrix. The feature vector of the Bayes vector is defined as $(D_1, D_2, D_3 \ldots D_m)^T$, the D_i is the decision made by the ith classifier, and the decision of the Bayes classifier is taken from a finite set $V(v_1, v_2 \ldots v_n)$.

The prior probability that the *i*th classifier make decision *k* and the real class is *j* is defined as Eq. 13.

$$P(R=j|D_i=k)=\frac{1}{m}\sum_{i=1}^{m}\begin{cases}1 & \text{if } D_i=k \text{ and } R=j\\ 0 & \text{otherwise}\end{cases} \tag{13}$$

The variable m describes the number of the samples; then, we can deduce the post probability of the decision *j*, that the *i*th classifier make decision *k*, defined by Eq. 14:

$$P(D_i=k|R=j)=\frac{P(R=j|D_i=k)}{\sum_{l=1}^{n}P(R=j|D_i=l)}. \tag{14}$$

The advisable number of decisions that the classifier can make is defined by the variable n. With this post probability for every individual classifier, the Bayes classifier can make decisions based on a multinomial model [24]. The naive Bayes classifier uses the simplifying assumptions that the attribute values are conditionally independent and that individual classifiers are independent in this model. The probability of observing the conjunction of classifiers $D_1, D_2, D_3 \ldots D_m$ is just the product of the post probability of the individual classifiers, in which case, our Bayes classifier will make a decision based on Eq. 15.

$$v=\underset{v_l\in V}{\operatorname{argmax}}P(R=l)\prod_{i=0}^{m}P(R=l|D_i) \tag{15}$$

The key structure of the naive Bayes fusion kernel in this model is the post probability matrix for individual classifiers that we trained with labeled samples. To adjust our model to fit the changed real-world model, we changed the post probability matrix.

As depicted in Fig. 2, to evolve the Bayes kernel, a new post probability matrix was trained to replace the old one with the labeled samples from the changed production line. The new classifier model is combined by the naive Bayes kernel with a new post probability matrix and SVM subclassifiers. The Bayes classifier takes advantage of not only the true positive result but also the true negative result from the subclassifiers. Because some subclassifiers may lose efficacy after a real-world model change and make a biased classification, this message can also be utilized by the integrated classifier.

6 Experimental results

To evaluate the effectiveness of the inspection system for surface defects, a surface defect data set was used. We then compared this approach with some other classification methods. In addition, some factors were examined that demonstrate how they affected classification accuracy.

6.1 Experiment implementation details

The accuracy of this evolutionary classifier has been compared with other classifiers such as SVM [25], NN-BP [26], and KNN [3]. To reveal the fairness of the classifier, a surface defect database, the NEU surface defect database[1] [6] was used. There are six kinds of typical defects of the hot-rolled steel strip surface in the database and 1800 gray-scale images, with 300 samples for every defect: rolled in scale (RS), crazing (CR), inclusion (IN), patches (PA), scratches (SC), and pitted surface (PS). Defect images collected and sampled at resolution are presented in Fig. 6.

The bias between different production lines is mainly caused by electronic circuit noise and sensor noise owing poor illumination and/or high temperature and these factors often lead to Gaussian noise in image acquisition [27], so we added Gaussian noise to the NEU database with different variances to simulate the defect images from different changed production lines. Fourteen contrasting data sets are used with standard deviations from $(0-13)$, the data with 0 deviation are the original data set. Paired photographs are shown in Fig. 7.

6.2 Adaptiveness of the classifier

To evaluate the adaptiveness of this evolutionary integrated classifier, the original NEU defect data set and 13 defect data sets formed by adding noise to the NEU defect data set were used. The standard deviations of the noise added to the NEU defect set were used (0–13), with an equal mean of 0. A total of 212 features were extracted from every defect image.

A BYEC classifier composed of 25 SVM subclassifiers was trained by 70% of the original NEU defect data set, and the remaining 30% of the data were used to evaluate the accuracy of the BYEC classifier on the original data set. Then, we randomly sampled 10% of the processed data sets to adjust our BYEC classifier. Finally, the accuracy of the adjusted BYEC classifier and the original classifier on the processed data set were tested by the remaining 90% of the processed data. As for the BYEC classifier, 70% of the original data set were used to train the KNN, BPNN, and SVM classifier and 30% were used to evaluate the accuracy on the original data set, then the accuracy of these classifiers on the processed data set was tested by 90% of the processed data set. The BPNN and KNN parameters were determined by cross-validation testing. The average accuracy of the classifiers on every data set was run 100 times, and the data sets were sampled individually.

Figure 8 depicts the accuracy obtained by using the KNN, BPNN, SVM, original BYEC, and BYEC classifiers. The standard deviation of the defect samples ranges from 0 to 13 and increases by 1 at each iteration, so the first data set is the original data set, the second has added

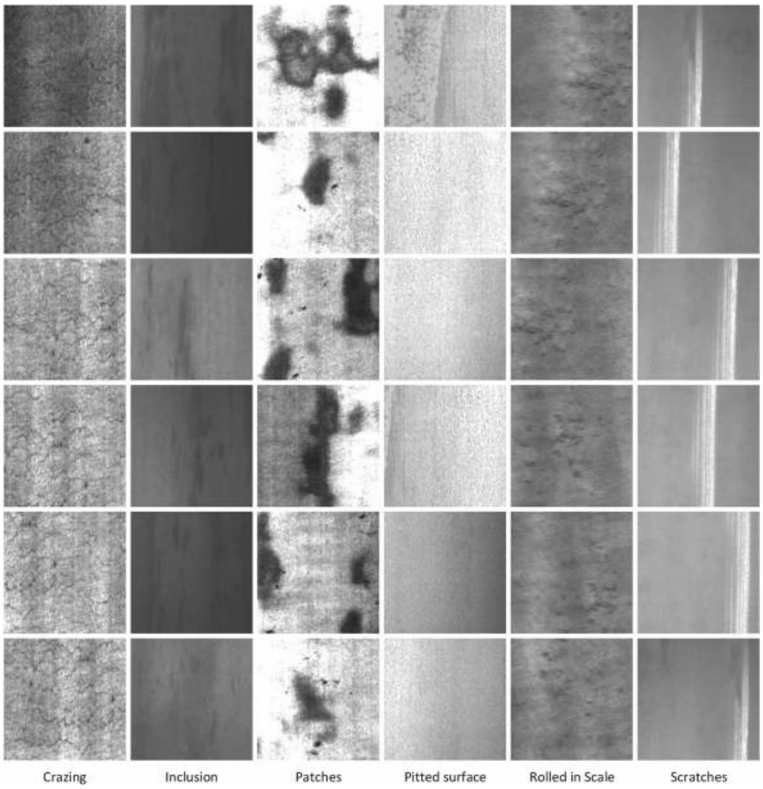


Fig. 6 Defect images, with *each row* being one of the six typical surface defects in the NEU database of sampling from 300 samples for one class

Gaussian noise with standard deviation 1, and so on. The accuracy associated with the SVM is higher than the corresponding values for the KNN, original BYEC, and BYEC classifiers for the original data set, but the accuracies of the SVM, KNN, BPNN, and original BYEC classifiers all decline as the standard deviation increases. The accuracies of the KNN, SVM, and BPNN classifiers decline by nearly 30%, and the original BYEC classifier declines a little more slowly than them. One possible reason for the slower decrease of the original BYEC classifier may be

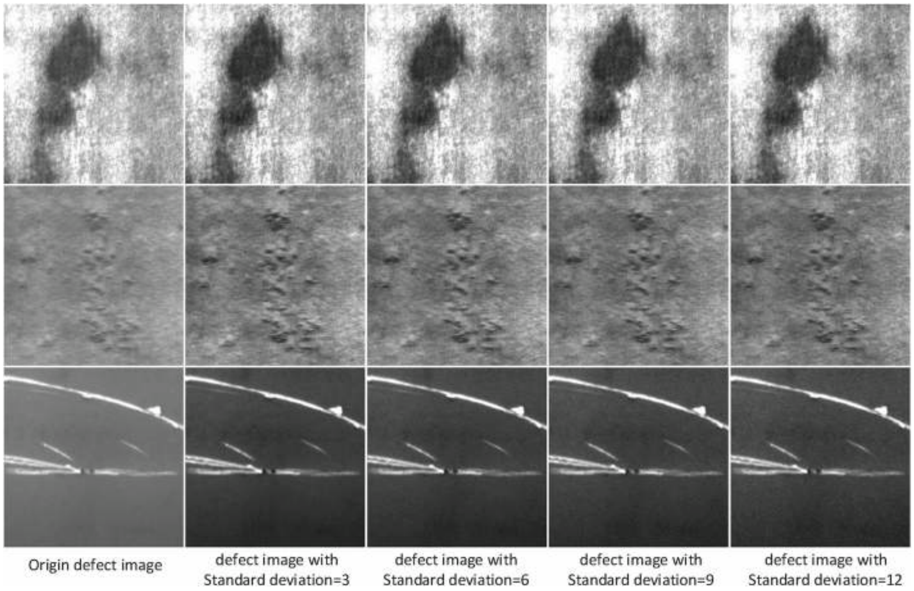
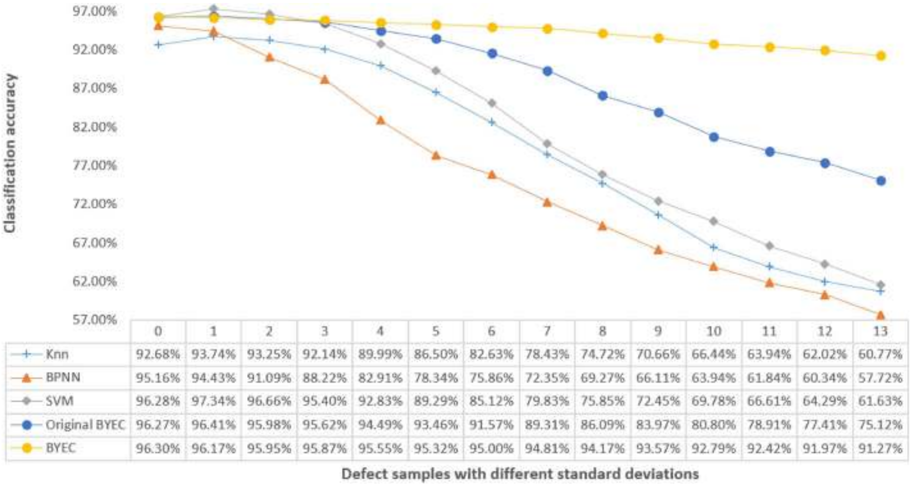


Fig. 7 Paired photographs of defects



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Knn	92.68%	93.74%	93.25%	92.14%	89.99%	86.50%	82.63%	78.43%	74.72%	70.66%	66.44%	63.94%	62.02%	60.77%
BPNN	95.16%	94.43%	91.09%	88.22%	82.91%	78.34%	75.86%	72.35%	69.27%	66.11%	63.94%	61.84%	60.34%	57.72%
SVM	96.28%	97.34%	96.66%	95.40%	92.83%	89.29%	85.12%	79.83%	75.85%	72.45%	69.78%	66.61%	64.29%	61.63%
Original BYEC	96.27%	96.41%	95.98%	95.62%	94.49%	93.46%	91.57%	89.31%	86.09%	83.97%	80.80%	78.91%	77.41%	75.12%
BYEC	96.30%	96.17%	95.95%	95.87%	95.55%	95.32%	95.00%	94.81%	94.17%	93.57%	92.79%	92.42%	91.97%	91.27%

Fig. 8 Accuracy of different classifiers on the defect samples with different standard deviations (0 − 13)

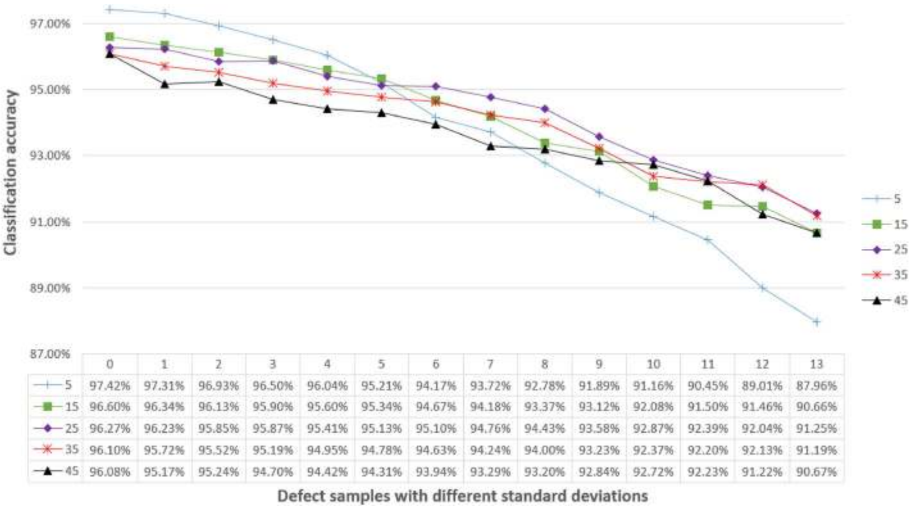
that more subclassifiers balance the bias from the original data set.

However, the increase of standard deviation has little impact on our BYEC classifier. It can be observed that this proposed method is more adaptive to different standard deviations in comparison with other classifiers. The accuracy of the BYEC classifier on the original data is lower than that of the SVM classifier, perhaps because of information loss from the combination of SVM classifiers. The original BYEC classifier without an adjustment process also suggests a relatively high adaptiveness compared with other classifiers.

6.3 Number of the sub-SVM classifiers

The number of sub-SVM classifiers, defined as k, is a key parameter for our BYEC classifier. It decides the particle size to which our system can be adjusted. The purpose of this section is to examine how k affects the accuracy and the adaptiveness of the BYEC classifier.

We trained five BYEC classifiers with k=5, 15, 25, 35, and 45. These classifiers were trained as described in Section 6.2 with the original defect set, and they were adjusted with 10% of the processed data and tested with 90% of the data set for evaluation. The experimental results are presented in Fig. 9. The highest accuracy on the original defect set is acquired by the classifier with $k = 5$, but the accuracy drops more rapidly compared with the other classifiers that perform at lower accuracy on the defect set with highest noise. The classifier with $k = 25$ gives the third highest accuracy compared to the original data set but acquires the highest accuracy on the defect set with the highest bias to the original data set. This figure



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
5	97.42%	97.31%	96.93%	96.50%	96.04%	95.21%	94.17%	93.72%	92.78%	91.89%	91.16%	90.45%	89.01%	87.96%
15	96.60%	96.34%	96.13%	95.90%	95.60%	95.34%	94.67%	94.18%	93.37%	93.12%	92.08%	91.50%	91.46%	90.66%
25	96.27%	96.23%	95.85%	95.87%	95.41%	95.13%	95.10%	94.76%	94.43%	93.58%	92.87%	92.39%	92.04%	91.25%
35	96.10%	95.72%	95.52%	95.19%	94.95%	94.78%	94.63%	94.24%	94.00%	93.23%	92.37%	92.20%	92.13%	91.19%
45	96.08%	95.17%	95.24%	94.70%	94.42%	94.31%	93.94%	93.29%	93.20%	92.84%	92.72%	92.23%	91.22%	90.67%

Fig. 9 Accuracy of BYEC classifiers with different k values (5, 15, 25, 35, and 45) on the defect samples with different standard deviations (0 − 13)

suggests that the BYEC classifier is more adaptive with a larger *k* value, but lower accuracy will be achieved with the BYEC classifier on the original data set.

As the results indicate in Fig. 9, *k* should be set with the real production environment. The BYEC classifier should be set with a larger *k* value when higher adaptive performance is needed. However, to avoid information loss, we should set the BYEC classifier with a smaller *k* value when the changed model has a small bias with the original model.

6.4 Set size of the sub-SVM classifiers

An important characteristic of classifiers is the size of the sample set that used to train the classifier. The more samples that are supplied, the more information the classifier machine can learn about the model. For our BYEC classifier, the accuracy of the subclassifiers can be evaluated more precisely with more samples.

The Fig. 10 depicts the accuracy of BYEC classifiers adjusted by different sizes of the samples. All the classifier were performed on the data set described at Section 6.2. We can see that the classifiers trained by 10% reach a lowest accuracy and have least adaptive, but the classifiers trained by 30% reached fairly good performance that the classifiers with more samples adjusted have little advantage over this classifier on accuracy and adaptive. Even in the defect set with standard deviation 13, the gap between the BYEC classifier trained by 10 and 50% is 1.01%. This illustrated that our BYEC converged very fast and the low requirement for the sample size.

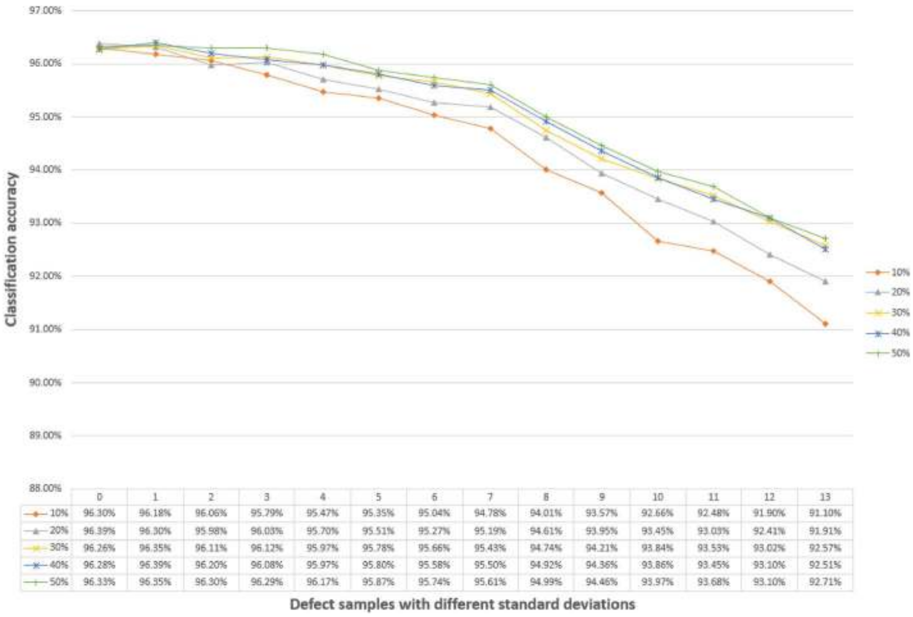
Figure 10 depicts the accuracy of BYEC classifiers adjusted by different sizes of the samples. All the classifiers were used on the data set described in Section 6.2. We can see that the classifiers trained by 10% reach the lowest accuracy and are the least adaptive, but the classifiers trained by 30% reach fairly good performance and classifiers with more samples adjusted have little advantage over this classifier in terms of accuracy and adaptiveness. Even in the defect set with a standard deviation of 13%, the gap between the BYEC classifier trained by 10% and that trained by 50% is only 1.01%. This illustrates that our BYEC classifier converged very fast and the low requirement for sample size.

6.5 Evaluating the effect of features

The five types of features selected are able to capture the properties of texture, color, and shape, respectively. Those features are described in detail by Neogi and proved to be very important for classifying steel defects [1]. To evaluate the effect of each feature, when the number of features is less than four, we observe that the accuracy of our method dropped very significantly, with accuracy barely reaching 70%. Therefore, we discuss the meaningful situation in which four features are used to classify the steel.

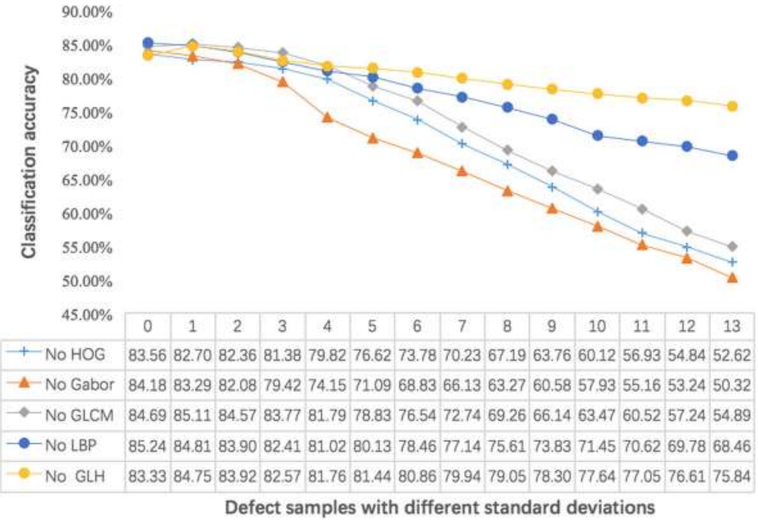
The Fig. 11 shows the accuracy of EFIC classifiers adjusted by different features. All the classifiers were used on the data set described at Section 6.2.

As shown in Fig. 11, the accuracy of EFIC classifiers without the Gabor feature is the lowest. However, the



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
10%	96.30%	96.18%	96.06%	95.79%	95.47%	95.35%	95.04%	94.78%	94.01%	93.57%	92.66%	92.48%	91.90%	91.10%
20%	96.39%	96.30%	95.98%	96.03%	95.70%	95.51%	95.27%	95.19%	94.61%	93.95%	93.45%	93.03%	92.41%	91.91%
30%	96.26%	96.35%	96.11%	96.12%	95.97%	95.78%	95.66%	95.43%	94.74%	94.21%	93.84%	93.53%	93.02%	92.57%
40%	96.28%	96.39%	96.20%	96.08%	95.97%	95.80%	95.58%	95.50%	94.92%	94.36%	93.86%	93.45%	93.10%	92.51%
50%	96.33%	96.35%	96.30%	96.29%	96.17%	95.87%	95.74%	95.61%	94.99%	94.46%	93.97%	93.68%	93.10%	92.71%

Fig. 10 Accuracy of BYEC classifiers adjusted by different sizes (0.1, 0.2, 0.3, 0.4, 0.5, and 0.6) on the defect samples with different standard deviations (0 − 13)



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
No HOG	83.56	82.70	82.36	81.38	79.82	76.62	73.78	70.23	67.19	63.76	60.12	56.93	54.84	52.62
No Gabor	84.18	83.29	82.08	79.42	74.15	71.09	68.83	66.13	63.27	60.58	57.93	55.16	53.24	50.32
No GLCM	84.69	85.11	84.57	83.77	81.79	78.83	76.54	72.74	69.26	66.14	63.47	60.52	57.24	54.89
No LBP	85.24	84.81	83.90	82.41	81.02	80.13	78.46	77.14	75.61	73.83	71.45	70.62	69.78	68.46
No GLH	83.33	84.75	83.92	82.57	81.76	81.44	80.86	79.94	79.05	78.30	77.64	77.05	76.61	75.84

Fig. 11 The accuracy of BYEC classifiers adjusted by different features on the defect samples with standard deviations (0 − 13)

Gabor feature is the most suitable for texture representation and discrimination of steel defect classification.

In contrast, the accuracy of EFIC classifiers without GLH is the highest. This demonstrates that the GLH feature is the least suitable for steel defect classification. The main reason for this is that the GLH feature can only capture gray features but not texture and or shape. Other good performance features followed by four features no HOG, GLCM, and LBP.

In conclusion, in our experiments, the absence of any one feature of EFIC classifiers significantly reduced the accuracy. Therefore, by combining these five features, our method can obtain satisfactory accuracy for steel defect classification.

7 Conclusions

Because accuracy decreases in steel surface classification systems with a changed production line model, in this research, we propose an evolutionary method that can be adjusted with a small sample set to fit a changed model and maintain relatively high accuracy. First, to overcome information loss in the process of evolution, we proposed five kinds of features that cover texture, color, and shape, respectively. Second, random subspace SVM classifiers are proposed to conquer the overfitting problem and fit for adjustment. Then, we introduced a naive Bayes machine to fuse the results from SVM subclassifiers that suits the adjustment and requires a small sample set. Finally, we introduced a simple method to adjust the Bayes kernel. The experimental results indicate that the BYEC algorithm is more adaptive with changed steel surface defect data set compared with other algorithms. Our research suggests that the adaptiveness of the classifier is highly related to the parameter *k*; with the growth of *k*, the BYEC classifier shows a greater adaptiveness but, unfortunately, with some accuracy loss on the original data set. The small sample set requirement was shown to have been fulfilled from the experiment results.

With the advantages and disadvantages of the BYEC algorithm, in a new production line, we can use the original BYEC algorithm without any labeled samples on the changed model; with the growth of the sample set size, we can adjust the BYEC model to become more adaptive. A new classifier can be trained to replace the old classifier as the relatively low accuracy on large sample set. Our future work will focus on increasing the accuracy on both a large sample set and a changed production model. Meanwhile, more noise-robust methods can be combined with this method to increase the adaptiveness.

Endnote

[1] NEU surface defect database is the the Northeastern University (NEU) surface defect database, download link: http://faculty.neu.edu.cn/yunhyan/NEU_surface_defect_database.html.

Acknowledgements
We thank Dr. Qiaochuan cheng from the Department of Electronics and Information Engineering, Tongji University, and anonymous reviewers for their useful comments and language editing which have greatly improved the manuscript.

Funding
This work is supported by the National Nature Science Foundation of China (NSFC) (60771065, 51378365), Foundation of Shanghai Institute of Technology (YJ2017-5).

Authors' contributions
MX and MJ conceived and designed the study. MX, MJ, LX, and LY performed the experiments. MX and MJ wrote the paper. GL, MX, MJ, LX, and LY reviewed and edited the manuscript. All authors read and approved the manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details
[1]School of Computer Science and Information Engineering, Shanghai Institute of Technology, Shanghai, China. [2]Department of Electronics and Information Engineering, Tongji University, Shanghai, China.

Received: 29 February 2016 Accepted: 4 July 2017
Published online: 17 July 2017

References

1. N Neogi, DK Mohanta, PK Dutta, Review of vision-based steel surface inspection systems. EURASIP J Image Video Process. **2014**(1), 1–19 (2014)
2. F Dupont, C Odet, M Cartont, Optimization of the recognition of defects in flat steel products with the cost matrices theory. NDT & E International. **30**(1), 3–10 (1997)
3. C Unsalan, A Erci, *Automated inspection of steel structures. Recent Advances in Mechatronics*. (Springer-Verlag Ltd, Singapore, 1999), pp. 468–480
4. S Ghorai, A Mukherjee, M Gangadaran, PK Dutta, Automatic defect detection on hot-rolled flat steel products. IEEE Trans. Instrum. Meas. **62**(3), 612–621 (2013). doi: 10.1109/TIM.2012.2218677
5. X-Y Wu, K Xu, J-W Xu, in *Image and Signal Processing, 2008. CISP'08. Congress on*. Application of undecimated wavelet transform to surface defect detection of hot rolled steel plates, vol. 4 (IEEE, 2008), pp. 528–532. http://ieeexplore.ieee.org/document/4566708/
6. K Song, Y Yan, A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects. Appl. Surface Sci. **285**, 858–864 (2013)
7. C Yan, Y Zhang, J Xu, F Dai, L Li, Q Dai, F Wu, A highly parallel framework for HEVC coding unit partitioning tree decision on many-core processors. IEEE Signal Process. Lett. **21**(5), 573–576 (2014)
8. G Wu, H Kwak, S Jang, K Xu, J Xu, in *Automation and Logistics, 2008 ICAL 2008. IEEE International Conference on*. Design of online surface inspection system of hot rolled strips (IEEE, 2008), pp. 2291–2295. http://ieeexplore.ieee.org/document/4636548/
9. Y-J Liu, J-Y Kong, X-D Wang, F-Z Jiang, in *Advanced Computer Theory and Engineering (ICACTE) 2010 3rd International Conference on*. Research on image acquisition of automatic surface vision inspection systems for steel sheet, vol. 6 (IEEE, 2010), pp. V6–189. http://ieeexplore.ieee.org/document/5579393/
10. M Muehlemann, *Standardizing defect detection for the surface inspection of large web steel*. (Illumination Technologies Inc, 2000), pp. 1–9. http://www.il-photonics.com/cdv2/Illumination%20tech-Light%20Sources/white%20papers/surface_inspection.PDF
11. T Ojala, M Pietikäinen, T Mäenpää, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 971–987 (2002)
12. RM Haralick, K Shanmugam, IH Dinstein, Textural features for image classification. IEEE Trans. Syst. Man Cybern, (6), 610–621 (1973). http://ieeexplore.ieee.org/document/4309314/
13. VN Vapnik, V Vapnik, *Statistical learning theory*, vol. 1. (Wiley, New York, 1998)
14. M Woźniak, M Graña, E Corchado, A survey of multiple classifier systems as hybrid systems. Inf. Fusion. **16**, 3–17 (2014)
15. JZ Kolter, M Maloof, et al., in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. Dynamic weighted majority: a new ensemble method for tracking concept drift (IEEE, 2003), pp. 123–130. http://ieeexplore.ieee.org/document/1250911/
16. N Dalal, B Triggs, in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Histograms of oriented gradients for human detection, vol. 1 (IEEE, 2005), pp. 886–893. http://ieeexplore.ieee.org/document/1467360/
17. JJ Henriksen, *3d surface tracking and approximation using gabor filters*. (South Denmark University, 2007). https://www.yumpu.com/en/document/view/44234347/3d-surface-tracking-and-approximation-using-gabor-filters-covil
18. UH-G Kreßel, in *Advances in kernel methods*. Pairwise classification and support vector machines (MIT Press, 1999), pp. 255–268. http://dl.acm.org/citation.cfm?id=299108
19. C-W Hsu, C-J Lin, A comparison of methods for multiclass support vector machines. IEEE Trans. Neural Netw. **13**(2), 415–425 (2002)
20. C Cortes, V Vapnik, Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
21. M Göksedef, Ş Gündüz-Öğüdücü, Combination of web page recommender systems. Expert Syst. Appl. **37**(4), 2911–2922 (2010)
22. C Porcel, A Tejeda-Lorente, M Martínez, E Herrera-Viedma, A hybrid recommender system for the selective dissemination of research resources in a technology transfer office. Inf. Sci. **184**(1), 1–19 (2012)
23. C Cabral, M Silveira, P Figueiredo, Decoding visual brain states from fMRI using an ensemble of classifiers. Pattern Recognit. **45**(6), 2064–2074 (2012)
24. M Vangelis, L Androutsopoulos, P Georgios, in *CEAS 2006 Third Conference on Email and AntiSpam (CEAS 2006)*. Spam filtering with naive bayes-which naive bayes? (Mountain View, 2006). www2.aueb.gr/users/ion/docs/ceas2006_paper.pdf
25. Y-J Jeon, D-C Choi, JP Yun, C Park, SW Kim, in *Control, Automation and Systems (ICCAS) 2011 11th International Conference on*. Detection of scratch defects on slab surface (IEEE, 2011), pp. 1274–1278. http://ieeexplore.ieee.org/document/6106307/
26. M Yazdchi, M Yazdi, AG Mahyari, in *Digital Image Processing, 2009 International Conference on*. Steel surface defect detection using texture segmentation based on multifractal dimension (IEEE, 2009), pp. 346–350. http://ieeexplore.ieee.org/document/5190595/
27. RC Gonzalez, *Digital image processing*. (Pearson Education, India, 2009). http://web.ipac.caltech.edu/staff/fmasci/home/astro_refs/Digital_Image_Processing_3rdEd_truncated.pdf