Applied Soft Computing 11 (2011) 1439–1447

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Employing multiple-kernel support vector machines for counterfeit banknote recognition[☆]

Chi-Yuan Yeh, Wen-Pin Su, Shie-Jue Lee[*]

Department of Electrical Engineering, National Sun Yat-Sen University, Kaohsiung 804, Taiwan

ARTICLE INFO

Article history:
Received 7 October 2008
Accepted 28 April 2010
Available online 7 May 2010

Keywords:
Banknote recognition
Support vector machine
Balanced error rate
Multiple-kernel learning
Semi-definite programming

ABSTRACT

Finding an efficient method to detect counterfeit banknotes is an imperative task in business transactions. In this paper, we propose a system based on multiple-kernel support vector machines for counterfeit banknote recognition. A support vector machine (SVM) to minimize false rates is developed. Each banknote is divided into partitions and the luminance histograms of the partitions are taken as the input of the system. Each partition is associated with its own kernels. Linearly weighted combination is adopted to combine multiple kernels into a combined matrix. Optimal weights with kernel matrices in the combination are obtained through semi-definite programming (SDP) learning. Two strategies are adopted to reduce the amount of time and space required by the SDP method. One strategy assumes the non-negativity of the kernel weights, and the other one is to set the sum of the weights to be unity. Experiments with Taiwanese banknotes show that the proposed approach outperforms single-kernel SVMs, standard SVMs with SDP, and multiple-SVM classifiers.

1. Introduction

With the advance of digital imaging technologies, color scanners and laser printers make it increasingly easier to produce counterfeit banknotes with high resolution. The proliferation of counterfeit banknotes in circulation leads to profit loss of traders and banks. Therefore, finding an efficient method to detect counterfeit banknotes is an imperative and demanding task for business transactions in our daily life.

Several approaches have been proposed for counterfeit banknote recognition. Takeda et al. [1] proposed a mask optimization technique using genetic algorithms (GA) and neural networks to detect counterfeit banknotes. Frosini et al. [2] employed neural networks to develop a paper currency recognition and verification system. He et al. [3] presented one-class classifiers for counterfeit banknote recognition. Each banknote is divided into $m \times n$ partitions. An individual classifier is constructed for each partition, and then these classifiers are combined to make the final decision. In addition, GA is employed to find optimal values for m and n. However, this GA-based method is very time consuming. Ionescu and Ralescu [4] also proposed one-class classifiers for counterfeit banknote recognition. For each banknote, several specific regions are considered, and each region is divided into $m \times n$ partitions. In addition, fuzzy Hamming distance is used to measure the similarity between banknotes.

Support vector machines (SVMs) have been shown to be an effective tool for solving classification problems [5–11]. However, the practitioner has to determine the kernel function and the associated kernel hyperparameters in advance. Unsuitably chosen kernel functions or hyperparameters may lead to significantly bad performance. Thus, determining suitable kernel functions and hyperparameters becomes an important issue for the application of SVMs [12–14]. Most researchers use trial-and-error to choose proper values for the hyperparameters; this obviously takes a lot of efforts. In addition, using a single kernel may not be sufficient to solve a complex problem accurately. Several researchers have adopted multiple kernels to solve these problems successfully [15–22].

Lanckriet et al. [18] used a linear combination of matrices to combine multiple kernels; they integrated multiple-kernel learning with SVMs, and transformed the problem into a semi-definite programming (SDP) problem, which, as a convex optimization problem, has a global optimum. SDP can be solved efficiently by the interior-point method. Other efficient multiple-kernel learning algorithms include Bach et al. [15], Sonnenburg et al. [21], and Rakotomamonjy et al. [20]. These approaches deal with large-scale problems by iteratively using the SMO algorithm [23] to update Lagrange multipliers and kernel weights in turn. Although these methods are faster than SDP, they tend to suffer from local minimum traps. Multiple-kernel learning based on hyperkernels has also been studied in [19,24]. On the other hand, Crammer et al. [17]

☆ This work was supported by the National Science Council under the grant NSC 95-2221-E-110-055-MY2.

* Corresponding author at: Department of Electrical Engineering, No. 70, Lienhai Rd., Kaohsiung 80424, Taiwan.
E-mail address: leesj@mail.ee.nsysu.edu.tw (S.-J. Lee).

1568-4946/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2010.04.015

and Bennett et al. [16] use boosting methods to combine heterogeneous kernel matrices.

We propose a system based on multiple-kernel support vector machines for counterfeit banknote recognition. For counterfeit banknote recognition, a false positive is usually more harmful than a false negative, since counterfeit banknotes can cause a bigger financial loss if they are not detected. We develop a SVM architecture to favorably reduce false positives. Each banknote is divided into partitions and the luminance histograms of the partitions are taken as the input of the system. Each partition is associated with its own kernels. Linearly weighted combination is adopted to combine multiple kernels into a combined matrix. By applying multiple-kernel learning, optimal weights with kernel matrices in the combination are obtained through semi-definite programming (SDP) learning. The original SDP problem was formulated on transduction setting where the kernel matrix is created by using the training patterns and the testing patterns. The amount of time and space may grow rapidly as the quantity of data increases. Instead, we consider an induction setting by using only the training patterns to construct the kernel matrix and adopt two strategies to improve the performance of SDP without degrading the accuracy. One strategy assumes the non-negativity of the kernel weights, and the other one is to set the sum of the weights to be unity. Experiments with Taiwanese banknotes show that the proposed method outperforms single-kernel SVMs, standard SVMs with SDP, and multiple-SVM classifiers.

The rest of this paper is organized as follows. Section 2 briefly introduces SVMs and multiple-kernel learning. In Section 3, the proposed SVM architectures are presented. Section 4 describes the proposed SDP with induction for multiple-kernel learning. The system for counterfeit banknote recognition is described in Section 5. Experimental and comparison results are presented in Section 6. Finally, conclusion is given in Section 7.

2. Background

In this section, basic concepts about SVMs are introduced. Composition of kernel matrices and semi-definite programming for multiple-kernel learning are also briefly described.

2.1. Weighted SVMs

Given a set of training patterns, SVM [5,25–27] is a kernel method which finds the maximum margin hyperplane in feature space to separate the training patterns into two groups. To allow for the possibility of outliers in the dataset and to make the method more robust, some patterns need not be strictly and correctly classified by the hyperplane, but the misclassified patterns should be penalized. For this purpose, slack variables ξ_i are introduced to account for the misclassified patterns. The objective function and constraints of the problem can therefore be formulated as:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ \text{s.t.} \quad & y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, 2, \ldots, l, \end{aligned} \tag{1}$$

where l is the number of training patterns, $\mathbf{x}_i$ is the input vector of training pattern i, and C is a parameter which gives a tradeoff between maximum margin and classification error. In the above setup, $\phi : X \to F$ is a mapping from the input space to a feature space, F, where patterns are more easily separated. Note that $\langle \mathbf{w}, \phi(\mathbf{x})\rangle + b = 0$ is the hyperplane in the feature space acting as the decision boundary for the two groups $y_i = +1$ and $y_i = -1$.

For some classification problems, different training patterns may have different impacts on the decision of the separating hyperplane [28–32]. Weighted SVM which uses different penalty parameters in the SVM formulation have been proposed for solving these problems as follows:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}d_i\xi_i \\ \text{s.t.} \quad & y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, 2, \ldots, l \end{aligned} \tag{2}$$

where each ξ_i can be assigned with a penalty d_i, allowing different training patterns to make different contributions to the construction of the hyperplane. When a training pattern owns a bigger weight, it is more likely to be classified correctly. Eq. (2) can be transformed to the following dual form:

$$\begin{aligned} \omega(K) = \quad & \max_{\boldsymbol{\alpha}} \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & Cd_i \geq \alpha_i \geq 0, \quad i = 1, 2, \ldots, l, \\ & \sum_{i=1}^{l} y_i\alpha_i = 0, \end{aligned} \tag{3}$$

where $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle$ is a kernel function, calculating the inner product between $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$. By rewriting Eq. (3) in matrix form, we have

$$\begin{aligned} \omega(K) = \quad & \max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T\mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^T G(K)\boldsymbol{\alpha} \\ \text{s.t.} \quad & \boldsymbol{\alpha} \geq 0, \\ & C\mathbf{d} - \boldsymbol{\alpha} \geq 0, \\ & \boldsymbol{\alpha}^T\mathbf{y} = 0, \end{aligned} \tag{4}$$

where $G(K) = \text{diag}(\mathbf{y})K\,\text{diag}(\mathbf{y})$, and $\mathbf{d} = [d_1, \ldots, d_l]^T$. Throughout this paper, the radial basis function (RBF) kernel is adopted. That is,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2), \tag{5}$$

where γ is the width parameter of the RBF kernel.

2.2. Kernel fusion and weight learning

Early SVM-based methods used a single-kernel function, $k(\mathbf{x}, \mathbf{y}) \equiv \langle \phi(\mathbf{x}), \phi(\mathbf{y})\rangle$, to calculate the inner product between two images in the feature space F. In practice, the kernel function is characterized by a kernel matrix computed from the training patterns. A kernel matrix is a square matrix $K \in R^{l\times l}$ where each matrix entry measures the similarity between $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$. If a dataset has varying local distributions, using a single kernel may lose some information to cope with this varying distribution. Most researchers use the trial-and-error heuristic to choose the best hyperparameters, which obviously takes a lot of efforts. Kernel fusion can help to solve this problem [15–17]. A simple direct sum fusion can be defined as $k(\mathbf{x}_i, \mathbf{x}_j) \equiv \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle$, where Φ is a new feature mapping defined as $\Phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \ldots, \phi_M(\mathbf{x})]^T$. This feature mapping can handle issues of varying pattern distributions by using multiple-kernel functions. The kernel matrix can be easily written as $K = K_1 + \cdots + K_M$ in this case, with K_i obtained from ϕ_i. This simple fusion can be generalized to a weighted combination of kernel matrices as follows:

$$K = \sum_{s=1}^{M}\mu_s K_s, \tag{6}$$

where M is the total number of kernel matrices and μ_s is the weight of the sth kernel matrix. The goal of kernel fusion is to find an optimal weight (μ_s) for each kernel matrix. When optimal kernel weights are obtained, the optimal hyperplane will be located correspondingly.

The objective function and constraints of the multiple-kernel learning problem can be formulated as follows:

$$\begin{aligned} \min_{\tilde{K}} \quad & \max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T \mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^T G(K)\boldsymbol{\alpha} \\ \text{s.t.} \quad & \boldsymbol{\alpha} \ge 0, \\ & C\mathbf{d} - \boldsymbol{\alpha} \ge 0, \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \tilde{K} \succcurlyeq 0, \\ & \mathrm{trace}(\tilde{K}) = c, \end{aligned} \tag{7}$$

where $\tilde{K} = \sum_{s=1}^{M} \mu_s \tilde{K}_s$ is obtained from the training patterns and the testing patterns, and $K = \sum_{s=1}^{M} \mu_s K_s$ is obtained from the training patterns and $G(K) = G(\sum_{s=1}^{M} \mu_s K_s) = \mathrm{diag}(\mathbf{y})(K)\mathrm{diag}(\mathbf{y})$.

Lanckriet et al. [18] transformed Eq. (7) into the SDP standard form as follows:

$$\begin{aligned} \min_{t,\tilde{K},\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\ \text{s.t.} \quad & \tilde{K} \succcurlyeq 0, \\ & \mathrm{trace}(\tilde{K}) = c, \\ & \begin{bmatrix} G(K) & (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda \mathbf{y}) \\ (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda \mathbf{y})^T & t - 2C\boldsymbol{\delta}^T \mathbf{d} \end{bmatrix} \succcurlyeq 0, \\ & \boldsymbol{\nu} \ge 0, \\ & \boldsymbol{\delta} \ge 0, \end{aligned} \tag{8}$$

where $\boldsymbol{\nu} \ge 0$, $\boldsymbol{\delta} \ge 0$, and $\lambda \in R$ are Lagrange multipliers, and $A \succcurlyeq 0$ means that A is a positive semi-definite matrix. Eq. (8) can then be solved by applying the primal-dual interior-point method.

3. Proposed SVM architectures

The goal of counterfeit banknote recognition is to minimize the number of false decisions. A false decision can be a false positive (FP) or false negative (FN). A FP is a forged note that was recognized as a genuine note, while a FN is a genuine note that was recognized as a forged note. In particular, we would like to minimize FPs more preferably than to minimize FNs, since a FP can cause a bigger financial loss than a FN.

3.1. Single-kernel weighted SVM

The single-kernel SVM designed for our purpose can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\sum_{\{i|y_i=+1\}} d_i^+ \xi_i + \sum_{\{i|y_i=-1\}} d_i^- \xi_i\right) \\ \text{s.t.} \quad & y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b) \ge 1 - \xi_i,\ \xi_i \ge 0,\ i = 1, 2, \ldots, l \end{aligned} \tag{9}$$

where $y_i = +1$ denotes that $\mathbf{x}_i$ is a genuine banknote while $y_i = -1$ denotes that $\mathbf{x}_i$ is a counterfeit banknote. The penalty parameter d_i^+ and d_i^- of Eq. (2) are defined as

$$d_i^+ = \frac{1}{2l^+}, \tag{10a}$$

$$d_i^- = \frac{1}{2l^-} \tag{10b}$$

where l^+ is the number of genuine banknotes, and l^- is the number of counterfeit banknotes in the training set. Note that $l^+ + l^- = l$. Usually, l^+ is much larger than l^- since counterfeit banknotes are rarely found. Therefore, the penalty d^+ associated with a FP is larger than the penalty d^- associated with a FN. A dual form, which is identical to Eq. (4), can be derived in a similar way.

Interestingly, the penalty term in Eq. (9) is closely related to the balanced error rate (BER) which was adopted as the main performance index in NIPS 2003 Feature Selection Challenge [33]. BER is defined as

$$\mathrm{BER} = \frac{1}{2}(\mathrm{FNR} + \mathrm{FPR}) = \frac{1}{2}\left(\frac{FN}{TP + FN} + \frac{FP}{TN + FP}\right) = \frac{FN}{2l^+} + \frac{FP}{2l^-} \tag{11}$$

where TP is the number of true positives and TN is the number of true negatives for the training patterns. True positives and true negatives are notes that were recognized correctly. If we simplify ξ_i to 1, the penalty term in Eq. (9) would be

$$\begin{aligned} \sum_{\{i|y_i=+1\}} d_i^+ \xi_i + \sum_{\{i|y_i=-1\}} d_i^- \xi_i &= \sum_{\{i|y_i=+1\}} \frac{\xi_i}{2l^+} + \sum_{\{i|y_i=-1\}} \frac{\xi_i}{2l^-} \\ &= \frac{FN}{2l^+} + \frac{FP}{2l^-} = \mathrm{BER}. \end{aligned} \tag{12}$$

In this case, our proposed SVM can minimize the balanced error rate.

3.2. Multiple-kernel weighted SVM

Note that the first two constraints of Eq. (8) imply that the amount of time and space requirements grows rapidly as the quantity of data increases, since $\tilde{K}$ is obtained from the training patterns and the testing patterns. One needs to check whether the combined matrix is positive semi-definite in each iteration. Furthermore, the constraint for the weights is loose in the last constraint, i.e., μ_s, $s = 1, \ldots, M$ can be negative. This introduces huge search space for finding optimal solutions. To reduce the computational complexity, we consider an induction setting by using only the training patterns to construct the kernel matrix. In addition, we adopt two strategies to help narrow down the search space for kernel weights. The first strategy is to assume the non-negativity of kernel weights and the second strategy is to set the sum of weights equal to 1. The multiple-kernel optimization problem can thus be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\mu}} \quad & \max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T \mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^T G(K)\boldsymbol{\alpha} \\ \text{s.t.} \quad & \boldsymbol{\alpha} \ge 0, \\ & C\mathbf{d} - \boldsymbol{\alpha} \ge 0, \\ & \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \boldsymbol{\mu} \ge 0, \\ & \mathbf{e}^T \boldsymbol{\mu} = 1 \end{aligned} \tag{13}$$

where $K = \sum_{s=1}^{M} \mu_s K_s$ is obtained from the training patterns.

Let $\boldsymbol{\alpha}^*$ and $\boldsymbol{\mu}^*$ be the solutions to Eq. (13). Define $I_{sv} = \{i | i \in \{1, 2, \ldots l\}, \alpha_i^* > 0\}$. With $k \in I_{sv}$, the optimal offset (b^*) can be obtained by the following equation:

$$b^* = y_k - \sum_{i=1}^{l} \alpha_i^* y_i \left(\sum_{s=1}^{M} \mu_s^* k_s(\mathbf{x}_k, \mathbf{x}_i)\right). \tag{14}$$

Then the decision function obtained from the multiple-kernel SVM-based classifier can be expressed as

$$f(\mathbf{x}) = \mathrm{sgn}\left(\sum_{i=1}^{l} \alpha_i^* y_i \left(\sum_{s=1}^{M} \mu_s^* k_s(\mathbf{x}, \mathbf{x}_i)\right) + b^*\right) \tag{15}$$

where *sgn* is the sign function.

4. SDP with induction

By following the process proposed by Lanckriet et al. [18], we can transform Eq. (13) into the SDP standard form which can then be solved by applying the primal-dual interior-point method. A SDP problem is an optimization problem with a linear objective function, linear matrix inequality (LMI) constraints, and affine equality constraints. First of all, we introduce an auxiliary variable t that serves as an upper bound on the objective function, and Eq. (13) becomes:

$$\begin{aligned} \min_{t,\boldsymbol{\mu}} \quad & t \\ \text{s.t.} \quad & t \geq \max_{\boldsymbol{\alpha}\geq 0, C\mathbf{d}-\boldsymbol{\alpha}\geq 0, \boldsymbol{\alpha}^T\mathbf{y}=0} \boldsymbol{\alpha}^T\mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^T G(K)\boldsymbol{\alpha}, \\ & \boldsymbol{\mu} \geq 0, \\ & \mathbf{e}^T\boldsymbol{\mu} = 1. \end{aligned} \tag{16}$$

By the Lagrange technique, the maximum on the right hand side of the first constraint can be expressed with the following Lagrangian:

$$L(\boldsymbol{\alpha}, \boldsymbol{\nu}, \boldsymbol{\delta}, \lambda) = \boldsymbol{\alpha}^T\mathbf{e} - \frac{1}{2}\boldsymbol{\alpha}^T G(K)\boldsymbol{\alpha} + \boldsymbol{\nu}^T\boldsymbol{\alpha} + \boldsymbol{\delta}^T(C\mathbf{d} - \boldsymbol{\alpha}) + \lambda(\boldsymbol{\alpha}^T\mathbf{y}) \tag{17}$$

where $\boldsymbol{\nu} \geq 0$, $\boldsymbol{\delta} \geq 0$, and $\lambda \in R$ are Lagrange multipliers. L has to be maximized with respect to $\boldsymbol{\alpha}$ given $\boldsymbol{\nu}$, $\boldsymbol{\delta}$, and λ, and then minimized with respect to $\boldsymbol{\nu}$, $\boldsymbol{\delta}$, and λ.

Setting partial derivatives of L with respect to primal variables equal to 0 yields the following results:

$$\begin{aligned} & \frac{\partial L}{\partial \boldsymbol{\alpha}} = \mathbf{e} - G(K)\boldsymbol{\alpha} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y} = 0, \\ \Rightarrow \quad & G(K)\boldsymbol{\alpha} = \mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}, \\ \Rightarrow \quad & \boldsymbol{\alpha} = (G(K))^{-1}(\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}). \end{aligned} \tag{18}$$

Note that $G(K)$ may not be invertible for some combinations of the kernel weights. This can be overcome by using the following modification:

$$\boldsymbol{\alpha} = (G(K) + \varepsilon I)^{-1}(\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}) \tag{19}$$

where ε is an arbitrarily small positive real number, and I is a $l \times l$ identity matrix. Substituting Eq. (18) into Eq. (17), we can eliminate the primal variables $\boldsymbol{\alpha}$, turning the Lagrangian into the Wolfe dual form:

$$\omega(K) = (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y})^T(G(K))^{-1}(\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}) + 2C\boldsymbol{\delta}^T\mathbf{d}. \tag{20}$$

This implies that for any $t > 0$, the inequality constraint $t \geq \omega(K)$ holds if and only if there exits $\boldsymbol{\nu} \geq 0$, $\boldsymbol{\delta} \geq 0$, and λ such that

$$t \geq (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y})^T(G(K))^{-1}(\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}) + 2C\boldsymbol{\delta}^T\mathbf{d}. \tag{21}$$

Because $G(K) \succcurlyeq 0$ and $(t - 2C\boldsymbol{\delta}^T\mathbf{d}) - (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y})^T(G(K))^{-1}(\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}) \geq 0$, the inequality constraint is now equivalent to the following LMI due to the Schur complement lemma [34]:

$$\begin{bmatrix} G(K) & (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}) \\ (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y})^T & t - 2C\boldsymbol{\delta}^T\mathbf{d}. \end{bmatrix} \succcurlyeq 0. \tag{22}$$

Taking Eq. (22) into account, Eq. (16) can be expressed in the following standard SDP form:

$$\begin{aligned} \min_{t,\boldsymbol{\mu},\lambda,\boldsymbol{\nu},\boldsymbol{\delta}} \quad & t \\ \text{s.t.} \quad & \begin{bmatrix} G(K) & (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y}) \\ (\mathbf{e} + \boldsymbol{\nu} - \boldsymbol{\delta} + \lambda\mathbf{y})^T & t - 2C\boldsymbol{\delta}^T\mathbf{d} \end{bmatrix} \succcurlyeq 0, \\ & \boldsymbol{\mu} \geq 0, \\ & \boldsymbol{\nu} \geq 0, \\ & \boldsymbol{\delta} \geq 0, \\ & \mathbf{e}^T\boldsymbol{\mu} = 1. \end{aligned} \tag{23}$$

Then, by applying the primal-dual interior-point method, Eq. (23) can be solved. Finally, the decision function for classification can be obtained by Eqs. (14)–(15).

5. Recognition system

We propose a system based on our proposed SVMs for counterfeit banknote recognition. The block diagram of the system is shown in Fig. 1. In the training phase, the image of each training banknote, either genuine or forged, is captured (block 1). Then each image is divided into $m \times n$ partitions (block 2). The histograms of the partitions of all the training images are obtained (block 3) and used to form a combined kernel matrix (block 4). Each partition is associated with its own kernels. Optimal kernel weights and Lagrange multipliers are obtained, and the decision function is constructed by Eq. (15) (block 5). In the testing phase, our system is applied to determining whether a given banknote is genuine or forged (block 6).

We capture the image of a banknote by a webcam with the banknote placed on a backlight panel. Each pixel of an image is represented by three components: red (R), green (G), and blue (B). The dimension of an image is $1200 \times 520 \times 3 = 1,872,000$ pixels. The images of two Taiwanese banknotes, one genuine and the other counterfeit, are shown in Fig. 2. In Fig. 2(a), the watermark of a chrysanthemum and the figure '1000' can be seen easily. One can see the difference of watermarks between genuine and counterfeit banknotes. In addition, the difference in ink and paper quality between genuine and counterfeit banknotes can also be detected with the help of a backlight panel.

Note that the image obtained is large. Using the components of all the pixels as input is not practical for a classifier. Two methods are applied to reduce the input dimension. Firstly, we adopt the Y component of the YIQ coordinates of each pixel to represent the pixel. In the YIQ color space, Y represents the luminance information, while I and Q represent the chrominance information. A simple relationship exists between the RGB color space and the YIQ color space [35]. Experimental results show that the Y component works better than the other components. Secondly, we take the histogram of the Y components of all the pixels as the input for the classification system. We use 256 bins for each histogram. In this way, an input vector of size 256, instead of 1,872,000, is used

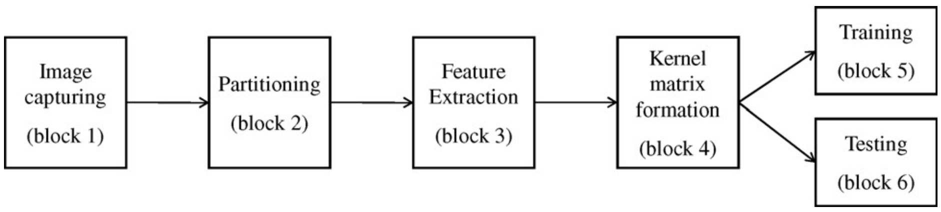


Image capturing (block 1)
Partitioning (block 2)
Feature Extraction (block 3)
Kernel matrix formation (block 4)
Training (block 5)
Testing (block 6)

Fig. 1. Block diagram of our system.

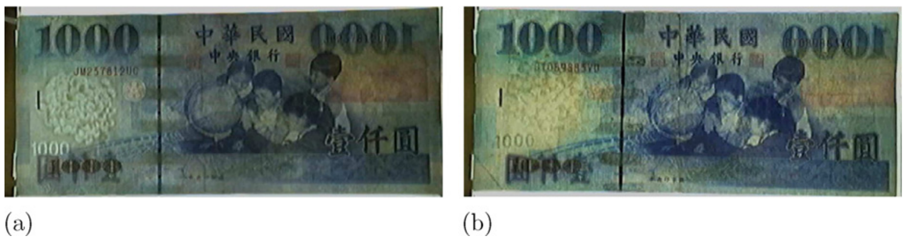


Fig. 2. Images of two banknotes taken from webcam. (a) Genuine banknote; (b) counterfeit banknote.

for each image. Fig. 3 shows the luminance histograms of the two banknotes of Fig. 2.

Using 256 histogram values to represent the image of a banknote may lose some information and reduce the discrimination power. To cope with varying histogram distributions in different areas of a banknote, each banknote is divided into $m \times n$ non-overlapping partitions of equal size, P_{ij}, $1 \le i \le m$, $1 \le j \le n$. For convenience, the partitions are labeled from left to right and then top to bottom. Each partition is represented by the luminance histogram of that partition. For example, the 2×2 partitions of a banknote and the histogram of each partition for a genuine banknote and a counterfeit banknote, respectively, are depicted in Fig. 4. Note that the black line near the center of the image is the window thread for the sake of imitation-prevention. From Figs. 3 and 4, we can see that, due to partitioning, the difference between the histograms of genuine and forged banknotes is more obvious.

Now we are ready to construct the combined kernel matrix for our system. Suppose each partition is associated with q RBF kernels having different widths. Let the combined kernel matrix be K. Then K is computed as

$$K = \sum_{s=1}^{M} \mu_s K_s \tag{24}$$

where $M = m \times n \times q$, and K_s is the kernel matrix of partition P_{ij} with the υth kernel, and

$$s = (i-1) \times n \times q + (j-1) \times q + \upsilon \tag{25}$$

with $1 \le \upsilon \le q$. Recall that each partition has an input vector of size 1×256. For a set of l training patterns, each K_s is a $l \times l$ matrix. The (a, b)th entry, $1 \le a, b \le l$, of K_s is $k(\mathbf{x}_a, \mathbf{x}_b)$ where k is the υth kernel associated with partition P_{ij}, and $\mathbf{x}_a$ and $\mathbf{x}_b$ are input vectors corresponding to partition P_{ij} of training patterns a and b.

Now we apply multiple-kernel learning to learn optimal weights for μ_s, $1 \le s \le M$, in Eq. (24). We construct a multiple-kernel weighted SVM according to Eq. (13). Then by applying SDP with induction, as described in Section 4, we can obtain $\boldsymbol{\alpha}^*$ and $\boldsymbol{\mu}^*$. The decision function for separating genuine from forged banknotes is then derived by Eqs. (14)–(15). This ends the training phase of our classification system. Now our system is ready to be applied to determining whether a given banknote is genuine or forged. The image of the given banknote is captured and partitioned. The histograms of the image substitute $\mathbf{x}$ in Eq. (15) (block 6). If the resulting function value is +1, then the given note is genuine. On the other hand, if the resulting function value is −1, then the given note is a forged one.

6. Experimental results

In this section, we test and compare the performance of the proposed method with other methods on a set of 99 Taiwanese banknotes. In these banknotes, 70 are genuine and 29 are forged. We randomly chose 50 genuine notes and 18 forged ones as training patterns, and the others as testing patterns. For convenience, we call the SVMs of Eq. (1) standard SVMs. In the following experiments, we use $C = 1$ for standard SVMs and $C = 2l^+$ for our proposed SVMs of Eq. (9). In addition, we use SeDuMi [36] as the SDP solver, and use YALMIP [37] to convert the equations into the standard form required by SeDuMi.

6.1. Experiment I

First of all, we compare the performance of our proposed SVMs with that of standard SVMs. A single kernel is used. No partitioning is done, i.e., the whole image of each banknote constitutes

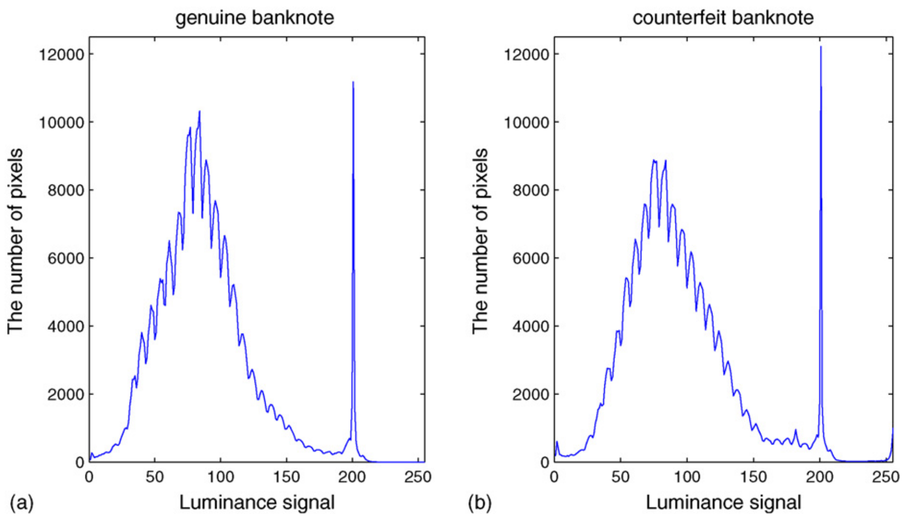
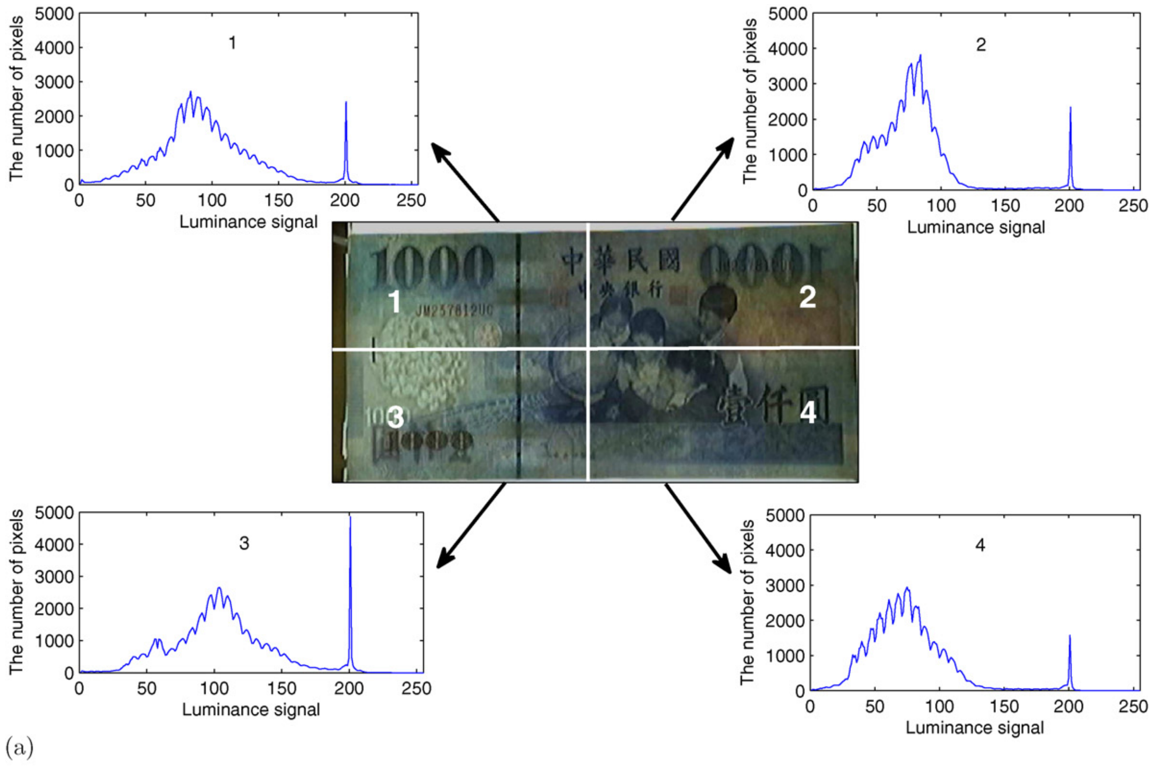


Fig. 3. Luminance histograms of two banknotes. (a) Genuine banknote; (b) counterfeit banknote.



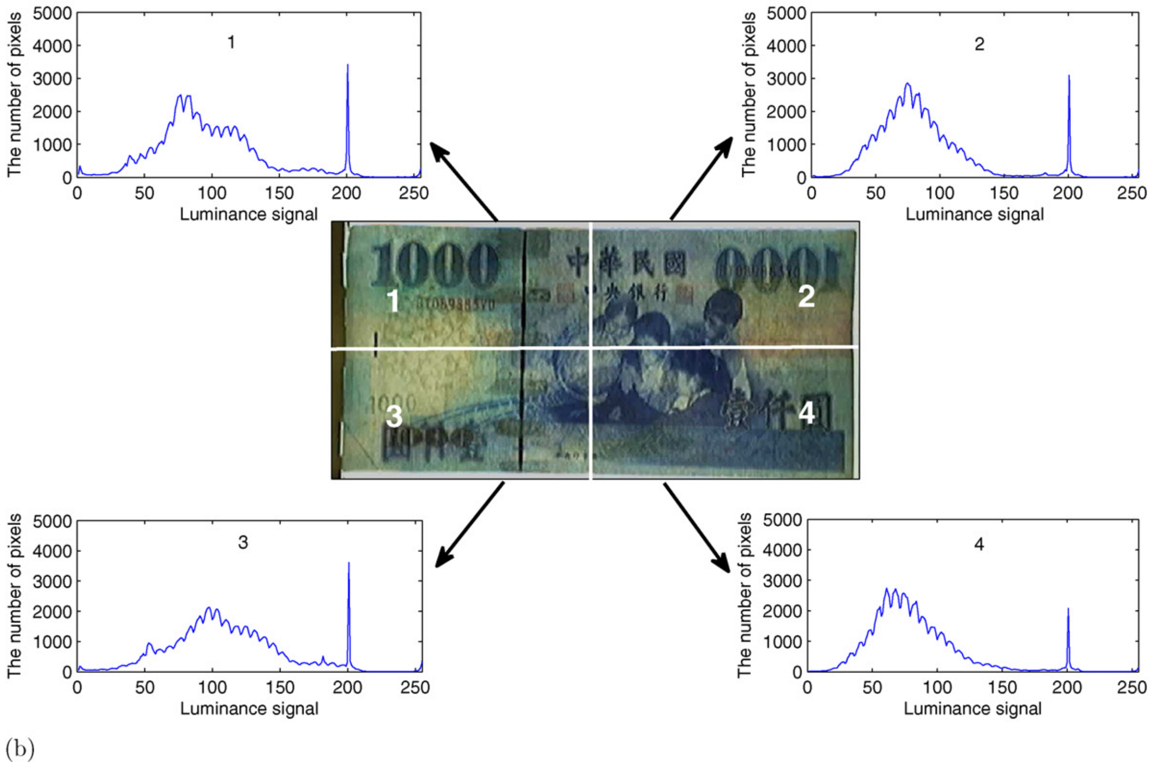


Fig. 4. Histograms of banknotes with 2 × 2 partitions. (a) Genuine banknote; (b) counterfeit banknote.

one partition. Table 1 presents the results obtained with different hyperparameters.

Note that in the table, Sd-Sk indicates that it is standard SVM with single kernel. Op-Sk stands for our proposed SVM with single kernel. FPR stands for false positive rate. FNR stands for false negative rate. ACC stands for accuracy. Best figures are indicated by bold fonts. From Table 1 we can see that our proposed SVMs performs better than standard SVMs. At hyperparameter $\gamma = 0.5$, both methods get 80.645% in accuracy. We also use line search to find the best hyperparameter $\gamma \in \{0.001, 0.002, \ldots, 10\}$. At $\gamma = 0.165$, Sd-Sk-Np achieves the highest accuracy rate of 83.871%, while at $\gamma = 0.2$, Op-Sk-Np

Table 1
Comparison between different SVM architectures (maximum values indicated in boldface).

Kernel	Sd-Sk ACC (%)/FPR (%)/FNR (%)	Op-Sk ACC (%)/FPR (%)/FNR (%)
$\gamma = 0.01$	67.742/90.909/0.000	70.968/45.455/20.000
$\gamma = 0.1$	74.194/54.546/10.000	77.419/36.364/15.000
$\gamma = 0.05$	74.194/54.546/10.000	74.194/45.455/15.000
$\gamma = 0.5$	**80.645** /45.455/5.000	**80.645** /45.455/5.000

achieves the highest accuracy rate of 87.097%. However, line search takes a lot of efforts.

6.2. Experiment II

In this experiment, we investigate the effect of using multiple kernels. No partitioning is done. The results are shown in Table 2 for two sets of kernels, 3-kernels and 5-kernels.

In the table, Op-Mk-Tr stands for our proposed SVM with multiple kernels and the kernel weights are learned by SDP with transduction, while Op-Mk-Id learns by SDP with induction. From Tables 1 and 2, we can see that using multiple kernels helps improve the accuracy rate, from 80.645% to 83.871%. However, the accuracy rate does not change noticeably with different sets of kernels. Note that our proposed approach improves the time efficiency of the original SDP method without degrading the performance.

6.3. Experiment III

In this experiment, we investigate the effect of applying the strategy of partitioning. Tables 3 and 4 present the results for two sets of kernels, $\gamma = [0.050.55]^T$ and $\gamma = [0.010.050.10.51]^T$, respectively.

In these tables, Op-Mk-Tr-Pa and Op-Mk-Id-Pa stand for Op-Mk-Tr and Op-Mk-Id, respectively, with partitioning. Clearly, different ways of partitioning may result in different performances. For example, partition 2×2 gets 100% in accuracy for Op-Mk-Id-Pa with 3-kernels, while partition 2×4 gets a poor performance, only 80.645% in accuracy. Also, a partition may behave differently with different sets of kernels. For example, partition 4×2 gets 96.774% in accuracy for Op-Mk-Id-Pa with 3-kernels, while it gets 100% in accuracy for Op-Mk-Id-Pa with 5-kernels. The variation in performance indicates the variation in histogram distribution due to different ways of partitioning. Again, Tables 3 and 4 show that our proposed SDP method improves the time efficiency of the original SDP method.

6.4. Experiment IV

In this experiment, we compare our recognition system with multiple-SVM classifiers [38–40]. In a multiple-SVM classifier, each banknote is divided into $m \times n$ partitions. A single-kernel SVM is constructed for each partition. Note that the same kernel is used for all partitions of a banknote. Then a posterior class probability $Pr(y_i = +1|\mathbf{x}^j)$ of the jth partition is calculated. The classifier output for class +1 is calculated by

$$Pr(y_i = +1|\mathbf{x}) \propto Pr(y_i = +1)^L \prod_{j=1}^{L} Pr(y_i = +1|\mathbf{x}^j) \tag{26}$$

where $L = m \times n$ and $Pr(y_i = +1) = \frac{l_+}{l}$. The classifier output for class −1 is calculated by

$$Pr(y_i = -1|\mathbf{x}) = 1 - Pr(y_i = +1|\mathbf{x}). \tag{27}$$

Table 2
Results obtained from combining multiple kernels.

RBF kernel	Op-Mk-Tr ACC (%)/FPR (%)/FNR (%)/time (s)	Op-Mk-Id ACC (%)/FPR (%)/FNR (%)/time (s)
$\boldsymbol{\gamma} = [0.050.55]^T$	**83.871** /36.364/5.000/15.074	**83.871** /36.364/5.000/10.646
$\boldsymbol{\gamma} = [0.010.050.50.11]^T$	80.645/45.455/5.000/15.960	**83.871** /36.364/5.000/13.046

Table 3
Results obtained from partitioning for $\gamma = [0.05\,0.5\,5]^T$.

RBF kernel	$\boldsymbol{\gamma} = [0.05\,0.5\,5]^T$	
Methods Partitions	Op-Mk-Tr-Pa ACC (%)/FPR (%)/FNR (%)/time (s)	Op-Mk-Id-Pa ACC (%)/FPR (%)/FNR (%)/time (s)
2×2	**96.774** /9.091/0.000/17.271	**100.000** /0.000/0.000/11.025
2×4	80.645/54.546/0.000/21.683	80.645/54.546/0.000/12.270
4×2	**96.774** /9.091/0.000/22.974	96.774/9.091/0.000/12.139
4×4	90.323/27.273/0.000/35.968	93.548/18.182/0.000/14.138
8×8	77.419/63.636/0.000/93.182	96.774/9.091/0.000/31.559
8×16	87.097/36.364/0.000/218.608	**100.000** /0.000/0.000/71.629

Table 4
Results obtained from partitioning for $\gamma = [0.01\,0.05\,0.1\,0.5\,1]^T$.

RBF kernel	$\boldsymbol{\gamma} = [0.01\,0.05\,0.1\,0.5\,1]^T$	
Methods Partitions	Op-Mk-Tr-Pa ACC (%)/FPR (%)/FNR (%)/time (s)	Op-Mk-Id-Pa ACC (%)/FPR (%)/FNR (%)/time (s)
2×2	93.548/18.182/0.000/18.806	**100.000** /0.000/0.000/11.815
2×4	87.097/36.364/0.000/24.418	80.645/54.546/0.000/13.403
4×2	**100.000** /0.000/0.000/26.111	**100.000** /0.000/0.000/13.656
4×4	93.548/18.182/0.000/39.462	93.548/18.182/0.000/16.878
8×8	80.645/54.546/0.000/177.823	90.323/27.273/0.000/53.398
8×16	90.323/27.273/0.000/497.102	93.548/18.182/0.000/149.829

Table 5
Results obtained by multiple-SVM classifiers.

RBF kernel	$\gamma = 0.05$	$\gamma = 0.5$	$\gamma = 5$
Partitions	ACC (%)/FPR (%)/FNR (%)	ACC (%)/FPR (%)/FNR (%)	ACC (%)/FPR (%)/FNR (%)
2 × 2	90.323/27.273/0.000	**93.548** /18.182/0.000	64.516/100.000/0.000
2 × 4	90.323/27.273/0.000	90.323/27.273/0.000	64.516/100.000/0.000
4 × 2	90.323/27.273/0.000	83.871/45.455/0.000	64.516/100.000/0.000
4 × 4	**93.548** /18.182/0.000	77.419/63.636/0.000	64.516/100.000/0.000
8 × 8	**93.548** /18.182/0.000	80.645/54.546/0.000	64.516/100.000/0.000
8 × 16	83.871/45.455/0.000	74.194/72.727/0.000	64.516/100.000/0.000

The posterior class probability $Pr(y_i = +1|\mathbf{x}^j)$ is approximated by a sigmoid function [38,39]:

$$Pr(y_i = +1|\mathbf{x}^j) \approx Pr_{A,B}(f_i) = \frac{1}{1 + exp(Af_i + b)} \tag{28}$$

where $f_i = f_i(\mathbf{x}^j)$.

Table 5 presents the results obtained by multiple-SVM classifiers for three different kernels.

We can see clearly that multiple-SVM classifiers provide worse performances than our system. In particular, multiple-SVM classifiers get very low accuracies for all ways of partitioning in the case of $\gamma = 5$. This shows the importance of choosing suitable hyperparameters for multiple-SVM classifiers.

7. Conclusion

We have presented a system based on multiple-kernel support vector machines for counterfeit banknote recognition. Each banknote is divided into partitions and the histograms of the partitions are taken as the input of the system. Each partition is associated with its own kernels. Linearly weighted combination is adopted to combine multiple kernels into a combined matrix. A SVM architecture, which allows false positive pattern to have a larger penalty than a false negative pattern was developed. Such SVMs can approximately minimize the balanced error rate. By applying multiple-kernel learning, optimal weights with kernel matrices in the combination are obtained through semi-definite programming (SDP) learning. We also proposed a modified multiple-kernel learning method which can narrow down the search space for searching optimal parameter settings. Only training patterns are used to construct kernel matrices, and we do not need to check whether the combined kernel matrix is positive semi-definite in each iteration. Experiments with Taiwanese banknotes show that the proposed method outperforms single-kernel SVMs, standard SVMs with SDP, and multiple-SVM classifiers.

Our system has one big advantage. Suppose more counterfeit-preventive features are added to the banknotes. Our system can still be capable of distinguishing between genuine and forged banknotes without any modification. Currently, only histograms of the captured images are used as the system input. Other features which are useful to increase the discrimination power of our system are being investigated. Furthermore, banknotes are easily contaminated due to their wide circulation. The contamination levels are most likely different for different banknotes. In addition, genuine banknotes may contain imperfections, and they may slightly different from one another. Fuzzy theory [41,42] can be a good technique to use for this problem.

References

[1] F. Takeda, T. Nishikage, S. Omatu, Banknote recognition by means of optimized masks, neural networks and genetic algorithms, Engineering Applications of Artificial Intelligence 12 (2) (1999) 175–184.
[2] A. Frosini, M. Gori, P. Priami, A neural network-based model for paper currency recognition and verification, IEEE Transactions on Neural Networks 7 (6) (1996) 1482–1490.
[3] C. He, M. Girolami, G. Ross, Employing optimized combinations of one-class classifiers for automated currency validation, Pattern Recognition 37 (6) (2004) 1085–1096.
[4] M. Ionescu, A. Ralusce, Fuzzy hamming distance based banknote validator, in: Proceedings of the 14th IEEE International Conference on Fuzzy Systems, 2005, pp. 300–305.
[5] C. Cortes, V. Vapnik, Support-vector network, Machine Learning 20 (3) (1995) 273–297.
[6] M. Pontil, A. Verri, Support vector machines for 3D object recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (6) (1998) 637–646.
[7] H. Drucker, D. Wu, V. Vapnik, Support vector machines for spam categorization, IEEE Transactions on Neural Networks 10 (5) (1999) 1048–1054.
[8] G. Guo, S.Z. Li, K. Chan, Face recognition by support vector machines, in: Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition, 2000, pp. 196–201.
[9] K.I. Kim, K. Jung, S.H. Park, H.J. Kim, Support vector machines for texture classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (11) (2002) 1542–1550.
[10] J.J. Ward, L.J. McGuffin, B.F. Buxton, D.T. Jones, Secondary structure prediction with support vector machines, Bioinformatics 19 (13) (2003) 1650–1655.
[11] Y. Hou, W. Hsu, M.L. Lee, C. Bystroff, Efficient remote homology detection using local structure, Bioinformatics 19 (17) (2003) 2294–2301.
[12] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Machine Learning 46 (1–3) (2002) 131–159.
[13] K. Duan, S. Keerthi, A.N. Poo, Evaluation of simple performance measures for tuning SVM hyperparameters, Neurocomputing 51 (4) (2003) 41–59.
[14] J.T.-Y. Kwok, The evidence framework applied to support vector machines, IEEE Transactions on Neural Networks 11 (5) (2000) 1162–1173.
[15] F.R. Bach, G.R.G. Lanckriet, M.I. Jordan, Multiple kernel learning, conic duality, and the SMO algorithm, in: Proceedings of the 21st International Conference on Machine Learning, 2004, pp. 6–13.
[16] K.P. Bennett, M. Momma, M.J. Embrechts, MARK: a boosting algorithm for heterogeneous kernel models, in: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 24–31.
[17] K. Crammer, J. Keshet, Y. Singer, Kernel design using boosting, in: S. Becker, S. Thrun, K. Obermayer (Eds.), Advances in Neural Information Processing Systems, vol. 15, MIT Press, Cambridge, MA, USA, 2003, pp. 537–544.
[18] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L.E. Ghaoui, M.I. Jordan, Learning the kernel matrix with semidefinite programming, Journal of Machine Learning Research 5 (2004) 27–72.
[19] C.S. Ong, A.J. Smola, R.C. Williamson, Learning the kernel with hyperkernels, Journal of Machine Learning Research 6 (2005) 1043–1071.
[20] A. Rakotomamonjy, F. Bach, S. Canu, Y. Grandvalet, More efficiency in multiple kernel learning, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 775–782.
[21] S. Sonnenburg, G. Rätsch, C. Schäfer, B. Schölkopf, Large scale multiple kernel learning, Journal of Machine Learning Research 7 (2006) 1531–1565.
[22] Z. Wang, S. Chen, T. Sun, MultiK-MHKS: a novel multiple kernel learning algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2) (2008) 348–353.
[23] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, USA, 1999, pp. 185–208.
[24] I.W.-H. Tsang, J.T.-Y. Kwok, Efficient hyperkernel learning using second-order cone programming, IEEE Transactions on Neural Networks 17 (1) (2006) 48–58.
[25] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, Cambridge, UK, 2000.
[26] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, Cambridge, MA, USA, 2001.
[27] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, Cambridge, UK, 2004.
[28] C.-F. Lin, S.-D. Wang, Fuzzy support vector machines, IEEE Transactions on Neural Networks 13 (2) (2002) 464–471.
[29] M. Wang, J. Yang, G.P. Liu, Z.J. Xu, K.C. Chou, Weighted-support vector machines for predicting membrane protein types based on pseudo-amino acid composition, Protein Engineering Design and Selection 17 (6) (2004) 509–516.
[30] Q. Tao, J. Wang, A new fuzzy support vector machine based on the weighted margin, Neural Processing Letters 20 (3) (2004) 139–150.

[31] Q. Tao, G.-W. Wu, F.-Y. Wang, J. Wang, Posterior probability support vector machines for unbalanced data, IEEE Transactions on Neural Networks 16 (6) (2005) 1561–1573.
[32] Y.-H. Liu, Y.-T. Chen, Face recognition using total margin-based adaptive fuzzy support vector machines, IEEE Transactions on Neural Networks 18 (1) (2007) 178–192.
[33] Neural information processing system conference, feature selection challenge (December 2003). http://www.nipsfsc.ecs.soton.ac.uk/.
[34] L. Vandenberghe, S. Boyd, Semidefinite programming, SIAM Review 38 (1) (1996) 49–95.
[35] J.C. Russ, The Image Processing Handbook, 5th edition, CRC Press, Boca Raton, FL, USA, 2006.
[36] J.F. Sturm, Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, Optimization Methods and Software 11 (1) (1999) 625–653.
[37] J. Löfberg, YALMIP: a toolbox for modeling and optimization in MATLAB, in: Proceedings of the 13th IEEE International Symposium on the Computer Aided Control Systems Design, 2004, pp. 284–289.
[38] J.C. Platt, Probabilistic outputs for support vector machines and comparison to regularized likelihood methods, in: A.J. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (Eds.), Advances in Large Margin Classifiers, MIT Press, Cambridge, MA, USA, 2000, pp. 61–74.
[39] H.-T. Lin, C.-J. Lin, R.C. Weng, A note on Platt's probabilistic outputs for support vector machines, Machine Learning 68 (3) (2007) 267–276.
[40] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley–Interscience, NJ, USA, 2004.
[41] L.A. Zadeh, Fuzzy sets, Information and Control 8 (3) (1965) 338–353.
[42] L.A. Zadeh, Knowledge representation in fuzzy logic, IEEE Transactions on Knowledge and Data Engineering 1 (1) (1989) 89–100.