Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 614543, 9 pages
http://dx.doi.org/10.1155/2013/614543

Research Article

Robust Template Decomposition without Weight Restriction for Cellular Neural Networks Implementing Arbitrary Boolean Functions Using Support Vector Classifiers

Yih-Lon Lin,[1] Jer-Guang Hsieh,[2] and Jyh-Horng Jeng[1]

[1] *Department of Information Engineering, I-Shou University, Kaohsiung 84001, Taiwan*
[2] *Department of Electrical Engineering, I-Shou University, Kaohsiung 84001, Taiwan*

Correspondence should be addressed to Jyh-Horng Jeng; jjeng@isu.edu.tw

Received 10 April 2013; Accepted 20 May 2013

Academic Editor: Ker-Wei Yu

If the given Boolean function is linearly separable, a robust uncoupled cellular neural network can be designed as a maximal margin classifier. On the other hand, if the given Boolean function is linearly separable but has a small geometric margin or it is not linearly separable, a popular approach is to find a sequence of robust uncoupled cellular neural networks implementing the given Boolean function. In the past research works using this approach, the control template parameters and thresholds are restricted to assume only a given finite set of integers, and this is certainly unnecessary for the template design. In this study, we try to remove this restriction. Minterm- and maxterm-based decomposition algorithms utilizing the soft margin and maximal margin support vector classifiers are proposed to design a sequence of robust templates implementing an arbitrary Boolean function. Several illustrative examples are simulated to demonstrate the efficiency of the proposed method by comparing our results with those produced by other decomposition methods with restricted weights.

1. Introduction

Cellular neural networks (CNNs) are large scale nonlinear circuits composed of locally connected cells, which was introduced in 1988 by Chua and Yang [1, 2]. CNN has a tremendous variety of applications in the fields of dynamic systems and signal processing [3–9]. The analysis of the dynamic behavior for the class of CNNs without feedback interconnections from neighboring cells, namely the uncoupled CNNs, is one of the popular research topics. A main feature of the uncoupled CNNs is that the binary steady state output in terms of the binary input of the CNN can be represented by a linearly separable Boolean function (LSBF) [1, 2].

In the study of uncoupled CNNs, most of the elementary applications can be derived and analyzed via Boolean functions, and this is directly related to the CNN template parameters. For LSBFs, Chen et al. [10] developed an essential relationship among the template, offset levels, a basis of the binary input vector set, and a neat truth table of the corresponding Boolean functions. In their work, they found a criterion for LSBFs and the criterion depends only on symbolic relations among the outputs of the Boolean functions. In [11, 12], the authors proposed an efficient method for implementing an LSBF and successfully realized all the 1882 and 94572 LSBFs by designing the corresponding CNN templates of 4 and 5 input variables, respectively. However, since the number of Boolean functions increases exponentially as the number of input variables increases, their method can be hardly extended.

The robustness indicator of a CNN is a measure quantifying the degree by which the parameters of a CNN can be perturbed while still producing the desired output. The definitions of robustness may be different in different contexts. Rigorous definitions of absolute and relative robustness of CNNs were given in [13]. A theoretical upper bound for relative robustness was derived and the absolute robustness can be arbitrarily increased by template scaling. CNN autoassociative memories were designed in [14, 15] using particle swarm optimization method with the robustness of designed memories taken into consideration.

The dynamic behavior of a class of third-order competitive CNNs depending on two parameters was investigated in [16]. In the special class of one-parameter family of symmetric CNNs, the authors discussed the robustness of the complete stability with respect to nonsymmetric perturbations on the neuron interconnections. In the VLSI implementation of CNN-UM (CNN Universal Machine), the template values will usually deviate from the ideal template values due to numerous reasons. Therefore robust design is crucial to guarantee correct outputs. To make the chip react as an ideal CNN structure, adaptive simulated annealing (ASA) method, a chip-specific optimization method, was proposed in [17] to automatically tune the template values. The result of the optimization process is the least sensitive template to the actual chip instance.

Recently, some machine learning techniques are applied to CNN applications. The recurrent fuzzy cellular neural network (RFCNN) was proposed for automatically constructing a multiple-CNN integrated neural system in [18–20]. The proposed RFCNN can automatically learn the network structure and the parameters simultaneously for uncoupled or coupled CNN. In the sense of learning, this structure learning includes the creation of fuzzy rules of fuzzy neural network and CNNs with pattern clustering algorithms. The parameter learning contains the tuning of fuzzy membership functions and CNN templates.

For an arbitrarily given Boolean function, Chen et al. [21, 22] proposed a generalization of Rosenblatt's perceptron model (universal perceptron) with DNA-like learning and decomposition algorithm. Their papers considered all of the Boolean functions via single-layer perceptron using the DNA-like learning algorithm. The proposed algorithms first train the DNA-like offset sequence and decompose the given linearly inseparable Boolean function into a sequence of LSBFs with logic XOR as conjunctions of the sequence of LSBFs. On the other hand, the CFC decomposition method, proposed by Crounse et al. [23], can be used to find a sequence of uncoupled CNNs implementing the given Boolean function, whether linearly separable or inseparable. In CFC method, the entries of the control templates of the required uncoupled CNNs are restricted to $\{0, \pm 1\}$ and the thresholds are some integers. The conjunctions of the sequence of CNNs are traditional logic operators. The method is a brute force one, yet it is simple and easy to implement.

It was pointed out in [24] that the geometric margin of a linear classifier with respect to a training data set, a notion borrowed from the machine learning theory, can conveniently be used to define the robustness of an uncoupled CNN implementing a linearly separable Boolean function. Larger geometric margin indicates better robustness against perturbations in both template parameters and the input data. Consequently, the so-called maximal margin classifiers (MMCs) can be devised to provide the most robust template design for uncoupled CNNs implementing linearly separable Boolean functions. An uncoupled CNN is said to be robust if it is a maximal margin classifier; that is, its template values are the solution of a maximal margin classification problem.

The decomposition method proposed by Lin et al. [24], abbreviated as LHJ method in this study, extends the CFC method in the sense that the entries of the control templates are restricted to more general sets $\{0, \pm 1, \pm 2\}$ or $\{0, \pm 1, \pm 2, \pm 3\}$, and all robust CNNs with template entries belonging to these weight-restricted constraint sets are characterized. Hence, under essentially the same search mechanism in CFC method, a much simpler search space, consisting of all possible robust CNNs with template values belonging to the weight-restricted set, is focused. The extension of the restricted weights above $\{0, \pm 1, \pm 2, \pm 3\}$ is computationally expensive because the number of searching templates will be an enormous number.

We wish to point out that the entries of the control template of a robust CNN are not required to be restricted to the set of small magnitude. For instance, the uncoupled CNN with the following control template and threshold:

$$w = \begin{pmatrix} 25 & 22 & 23 & 9 & 4 & 12 & 5 & 14 & 8 \end{pmatrix}, \qquad b = 25, \tag{1}$$

is robust. The purpose of this study is to find algorithms for robust template decomposition without weight restriction for CNNs implementing an arbitrary Boolean function via support vector classifiers (SVCs).

2. Uncoupled CNN

Consider a standard $M \times N$ CNN. In this study, we consider exclusively the most commonly used 3×3 neighborhood for each cell $C(i, j)$. Thus the CNN parameters may conveniently be represented by a triple (A, B, z), where A and B are 3×3 feedback and control templates, respectively, and z is the threshold value. In this study, we consider the following uncoupled CNN written as

$$A^0 = \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}, \quad B = \begin{array}{|c|c|c|} \hline w_1 & w_2 & w_3 \\ \hline w_4 & w_5 & w_6 \\ \hline w_7 & w_8 & w_9 \\ \hline \end{array}, \quad z = b. \tag{2}$$

With the static binary inputs, the steady-state output $y_{ij}(\infty)$ of $C(i, j)$ in this uncoupled CNN can be calculated explicitly as

$$y_{ij}(\infty) = \operatorname{sign}\left[\langle w, u \rangle + b\right], \tag{3}$$

where

$$w := \begin{bmatrix} w_1 & \cdots & w_9 \end{bmatrix}^T, \qquad u := \begin{bmatrix} u_1 & \cdots & u_9 \end{bmatrix}^T, \tag{4}$$

and $u_1, \ldots, u_9$ are inputs to the cell. See Theorem 6.1 in [25].

It is well known that a (local) Boolean function $h(u_1, \ldots, u_9)$ of nine variables is realizable by every cell of an uncoupled CNN if and only if $h(\cdot)$ can be expressed by the formula

$$h(u_1, \ldots, u_9) = \operatorname{sign}\left[a_1 u_1 + \cdots + a_9 u_9 + b\right], \tag{5}$$

where a_i, $i \in \underline{9}$, b are real constants, and $u_i \in \{1, -1\}$, $i \in \underline{9}$, is the ith Boolean variable. See Theorem 6.2 in [25]. It is important to note that the discriminant function

$$f(u) := a_1 u_1 + \cdots + a_9 u_9 + b \tag{6}$$

is an affine-linear function of $u \in \mathfrak{R}^9$. Thus implementing a Boolean function by an uncoupled CNN can be regarded as a linear classification problem.

3. Support Vector Classifiers

Since maximal margin and soft margin support vector classifiers will be utilized in our proposed decomposition methods, we briefly review the support vector classifiers in this section. For more background materials, see, for instance, [26–28] and the references therein.

Let $X \subseteq \mathfrak{R}^n$ and $Y := \{1, -1\}$. Suppose we are given the training set

$$S := \{(x_i, y_i)\}_{i=1}^{l} \subseteq X \times Y. \tag{7}$$

The training set S in (7) is said to be linearly separable if there is a hyperplane $H_{w,b}$ of the form

$$f_{w,b}(x) := \langle w, x \rangle + b = 0, \quad w \in \mathfrak{R}^n, \ b \in \mathfrak{R}, \tag{8}$$

that correctly classifies all training data. By treating the truth table of a given Boolean function as the training set with $l = 512$ training data, this training set must be linearly separable in order for the Boolean function to be realizable by an uncoupled CNN.

The geometric margin of the classifier $f_{w,b}(\cdot)$ with respect to the training set S is defined by [28]

$$\eta_S(w, b) := \min_{i=1}^{l} y_i \cdot \left[\left\langle \|w\|^{-1} w, x_i \right\rangle + \|w\|^{-1} b \right]. \tag{9}$$

This will be used as the robustness indicator of an uncoupled CNN implementing a linearly separable Boolean function [24]. Larger geometric margin indicates better robustness against perturbations in both template parameters and the input data.

Let the Boolean function be linearly separable. The maximal margin classifier (MMC) can be obtained by solving the following optimization problem:

$$\text{maximize} \quad \sum_{i=1}^{l} z_i - 2^{-1} \sum_{i=1}^{l} \sum_{j=1}^{l} z_i z_j y_i y_j \left\langle x_i, x_j \right\rangle \tag{10a}$$

$$\text{subject to} \quad \sum_{i=1}^{l} z_i y_i = 0, \quad z_i \geq 0 \ \forall i \in \underline{l}. \tag{10b}$$

Suppose (z^*, b^*) solves the problem in (10a) and (10b). Then the optimal weights are given by

$$w^* = \sum_{i=1}^{l} z_i^* y_i x_i. \tag{11}$$

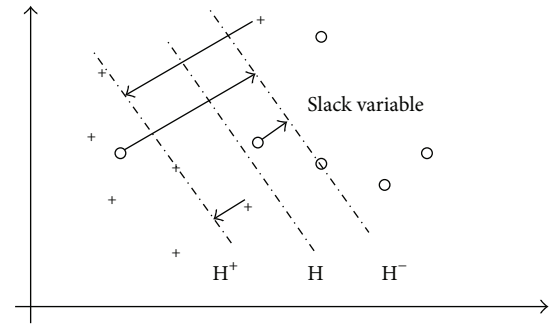

Figure 1: Slack variables for linear classification problems.

The optimal discriminant function is thus given by

$$f^*(x) = \langle w^*, x \rangle + b^* = \sum_{i=1}^{l} z_i^* y_i \langle x_i, x \rangle + b^*. \tag{12}$$

Using the KKT conditions of the optimization problem (10a) and (10b), it can be shown that the corresponding margin is given by [28]

$$\gamma_S = \|w^*\|^{-1} = \left(\sum_{i=1}^{l} z_i^* \right)^{-1/2}. \tag{13}$$

It is obvious that the maximal margin classification problem (10a) and (10b) has no solution if the training data of a Boolean function is linearly inseparable. Note that as the maximal margin is obtained, the outputs of the optimal discriminant function will retain the same positive/negative signs even when the template parameters and the input data are perturbed. Since the optimal separating hyperplane has the maximal geometric margin, the best robustness can be achieved for the designed CNN.

To allow for misclassifications of training data, we now introduce the slack variables for classification problems.

Let $\gamma > 0$ be given. The slack variable ξ_i of an example (x_i, y_i) with respect to the hyperplane $H_{w,b}$ and target margin γ is defined by [28]

$$\xi_i := \max\left(0, \gamma - y_i \cdot \left[\langle w, x_i \rangle + b \right]\right). \tag{14}$$

See Figure 1 for illustration. Clearly, ξ_i measures the amount by which the example (x_i, y_i) fails to have margin γ with respect to the hyperplane $H_{w,b}$. Consequently, the quantity defined by

$$\|\xi\|_1 := \sum_{i=1}^{l} \xi_i \quad \text{or} \quad \|\xi\|_2^2 := \sum_{i=1}^{l} \xi_i^2 \tag{15}$$

measures the total amount by which the training set fails to have margin γ and takes into account any misclassifications of the training data.

The 2-norm soft margin classifiers (SVC2) will be used in this study. The 1-norm soft margin classifiers (SVC1) may also be used. In soft margin classification problems, the target margin γ is set to the canonical value of 1.

The SVC2 can be obtained by solving the following optimization problem:

$$\text{maximize} \quad \sum_{i=1}^{l} z_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} z_i z_j y_i y_j \left\langle x_i, x_j \right\rangle - \frac{1}{2C}\sum_{i=1}^{l} z_i^2 \tag{16a}$$

$$\text{subject to} \quad \sum_{i=1}^{l} z_i y_i = 0, \qquad z_i \geq 0 \ \forall i \in \underline{l}. \tag{16b}$$

Here, the regularization (or smoothing) parameter $C > 0$ controls the tradeoff between the complexity of the machine and the number of nonseparable points.

Suppose (z^*, b^*) solves the problem in (16a) and (16b). Then the optimal weights and discriminant function are given by (11) and (12), respectively. Moreover, the slack variables ξ_i, $i \in \underline{l}$, are given by

$$\begin{aligned}\xi_i &:= \max\left(0, 1 - y_i \cdot \left[\left\langle w^*, x_i \right\rangle + b^*\right]\right) \\ &= \max\left(0, 1 - y_i \cdot \left[\sum_{j=1}^{l} y_j z_j^* \left\langle x_j, x_i \right\rangle + b^*\right]\right).\end{aligned} \tag{17}$$

4. Minterm and Maxterm CNNs

To understand our proposed decomposition algorithm, we first introduce the minterm and maxterm Boolean functions [24].

Definition 1. A Boolean function $\beta(u_1, \ldots, u_n)$ of n variables is said to be a minterm (resp., maxterm) if its output column in the truth table consists of all "−1"s (resp., "1"s) except for one entry having a "1" (resp., "−1").

Since there are 2^n rows in a truth table of n Boolean variables, for the usual case with $n = 9$ variables, there are $2^9 = 512$ distinct minterm (or maxterm) Boolean functions.

Every minterm (or maxterm) Boolean function is linearly separable [25]. In the truth table of a minterm (resp., maxterm) Boolean function, let $u_j^{\min}$ (resp., $u_j^{\max}$), $j \in \underline{n}$, be the values (−1 or 1) of the input variables corresponding to the row with output 1 (resp., −1). Then the optimal discriminant function with maximum (geometric) margin $1/\sqrt{n}$ is given by

$$\begin{aligned}f^{\min}(u) &= \operatorname{sign}\left(u_1^{\min}\right) u_1 + \cdots + \operatorname{sign}\left(u_n^{\min}\right) u_n - (n-1) \\ &\left(\text{resp., } f^{\max}(u) = -\operatorname{sign}\left(u_1^{\max}\right) u_1\right. \\ &\left. - \cdots - \operatorname{sign}\left(u_n^{\max}\right) u_n + (n-1)\right).\end{aligned} \tag{18}$$

For the usual case of $n = 9$, the optimal discriminant function with maximum (geometric) margin $1/\sqrt{9}$ is given by

$$\begin{aligned}f^{\min}(u) &= \operatorname{sign}\left(u_1^{\min}\right) u_1 + \cdots + \operatorname{sign}\left(u_9^{\min}\right) u_9 - 8 \\ &\left(\text{resp., } f^{\max}(u) = -\operatorname{sign}\left(u_1^{\max}\right) u_1\right. \\ &\left. - \cdots - \operatorname{sign}\left(u_9^{\max}\right) u_9 + 8\right).\end{aligned} \tag{19}$$

Table 1: Simple Boolean function.

	u_1	u_2	u_3	y
1	−1	−1	−1	−1
2	−1	−1	1	1
3	−1	1	−1	−1
4	−1	1	1	1
5	1	−1	−1	1
6	1	−1	1	1
7	1	1	−1	1
8	1	1	1	−1

A CNN which implements a minterm (resp., maxterm) Boolean function in each cell is called a minterm (resp., maxterm) CNN. Our proposed algorithm is based on the following realization theorem [25].

Theorem 2. *Every local Boolean function of n variables can be realized by ORing (in minterm decomposition) or ANDing (in maxterm decomposition) at most 2^n uncoupled CNNs.*

Let us consider for instance the Boolean function β with $n = 3$ given in the truth table shown in Table 1, where u_1, u_2, and u_3 are input variables and y is the output variable.

Table 1 can also be conveniently represented by the following figure by listing only the output values, where black and white dots represent logic values 1 and −1, respectively:

Since there are five 1 s in the output, according to the realization theorem, the Boolean function β can be decomposed by ORing 5 minterm CNNs:

$$\beta = \beta_2 \vee \beta_4 \vee \beta_5 \vee \beta_6 \vee \beta_7, \tag{20}$$

where β_j is a minterm Boolean function, having a "1" in the jth output and "−1"s elsewhere, and "∨" is the OR logic operator. It is desirable to shorten the length of the minterm decomposition. A natural idea is to group some minterms so that the resulting Boolean function is still linearly separable. Furthermore, we require that the uncoupled CNN realizing the resulting Boolean function is robust in the sense that it is a maximal margin classifier. For instance, the following decomposition produces exactly the same Boolean function:

$$\beta = \beta_1^{\min} \vee \beta_2^{\min} = \left(\beta_5 \vee \beta_6 \vee \beta_7\right) \vee \left(\beta_2 \vee \beta_4\right). \tag{21}$$

The first and second combined Boolean functions $\beta_1^{\min}$ and $\beta_2^{\min}$ are both linearly separable, which can be realized by the following robust uncoupled CNNs with control templates and thresholds, respectively:

$$\begin{aligned} w &= \begin{pmatrix} 2 & -1 & -1 \end{pmatrix}, \qquad b = -1, \\ w &= \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}, \qquad b = -1. \end{aligned} \tag{22}$$

The problem is how we group appropriate minterms.

5. Decomposition Algorithms

In this section, we describe our proposed minterm- and maxterm-based decomposition methods. Before proceeding, we give some motivation for the minterm-based decomposition algorithm. First notice that the maximal margin classification problem (10a) and (10b) has no solution if the training data of a Boolean function is not linearly separable. Suppose now we are given a general Boolean function. To find the first template in our proposed algorithm, we may start from the working Boolean function which is the given Boolean function β, now thought of as the minterm decomposition. Since we do not know whether the Boolean function β is linearly separable or not, we may solve (16a) and (16b) for an SVC2 with a guaranteed solution. This results in an optimal separating hyperplane but some data may be misclassified. Let I_{remain} be the set of all indices of data with output value 1 in the working Boolean truth table. If there are no misclassified data, then the algorithm stops because the working Boolean function is linearly separable and we may find the corresponding robust CNN by solving the maximal margin classification problem (10a) and (10b). Otherwise suppose that there are misclassified data, and the jth data with $j \in I_{\text{remain}}$ achieves the maximal value of slack variables among all data with indices in I_{remain}. The key idea is that this data might represent the point which is rather difficult to be correctly classified by the current hyperplane. In this case, the minterm with output value 1 in the jth data will be dropped from the working Boolean function. This process is repeated until we find a linearly separable working Boolean function or end up with a minterm which is certainly linearly separable.

The preceding discussion can be summarized as the following top-down (or pruning) algorithm which is based on the minterm realization theorem. Note that the entries of the Boolean truth table are either +1 or −1.

Algorithm 1 (minterm-based decomposition)

Data. A Boolean truth table consisting of $l = 2^n$ training data $S := \{(x_i, y_i)\}_{i=1}^{l} \subseteq \{+1, -1\}^n \times \{+1, -1\}$ with n predictors and one output.

Goal. A sequence of robust templates implementing the given Boolean truth table.

Step 1. Choose the regularization parameter $C > 0$ and the desired upper bound α, $0 \le \alpha \le 1$, for the slack variables ξ_i.

Step 2. Define

$$D := \{y_i\}_{i=1}^{l}, \quad I_{\text{remain}} \longleftarrow I_{\text{index}}, \quad m \longleftarrow 0. \tag{23}$$

Step 3. Define

$$I_{\text{index}} := \{i \in \underline{l} : y_i = 1\}. \tag{24}$$

Step 4. If $I_{\text{index}} = \varnothing$, go to Step 9, else go to Step 5.

Step 5. If $\#(I_{\text{remain}}) = 1$, say $j \in I_{\text{remain}}$, set $m \leftarrow m + 1$. Then the mth robust template is a minterm.

Step 5.1. Form the new target set D by setting $y_j = -1$.

Step 5.2. $I_{\text{index}} \leftarrow I_{\text{index}} \setminus I_{\text{remain}}$, $I_{\text{remain}} \leftarrow I_{\text{index}}$, and go to Step 3.

Step 6. If $\#(I_{\text{remain}}) > 1$, from the current l training data with target set D, solve the soft margin classification problem. Define

$$\begin{aligned} I_{\text{miss}} &:= \{i \in \underline{l} : \xi_i \ge \alpha\} \quad \text{if } \alpha = 1; \\ I_{\text{miss}} &:= \{i \in \underline{l} : \xi_i > \alpha\} \quad \text{if } \alpha \ne 1. \end{aligned} \tag{25}$$

Step 7. If $I_{\text{miss}} = \varnothing$, set $m \leftarrow m + 1$. Then the mth robust template is obtained by solving the maximal margin classification problem with target set D.

Step 7.1. Form the new target set D by setting $y_j = -1$ for all $j \in I_{\text{remain}}$.

Step 7.2. $I_{\text{index}} \leftarrow I_{\text{index}} \setminus I_{\text{remain}}$, $I_{\text{remain}} \leftarrow I_{\text{index}}$, and go to Step 3.

Step 8. If $I_{\text{miss}} \ne \varnothing$, find

$$I_{\text{temp}} = \arg\max_{i \in I_{\text{remain}}} \{\xi_i\}. \tag{26}$$

Step 8.1. If $\#(I_{\text{temp}}) < \#(I_{\text{remain}})$, set $I_{\text{delete}} = I_{\text{temp}}$; else choose any $j \in I_{\text{temp}}$ and set $I_{\text{delete}} = I_{\text{temp}} \setminus \{j\}$.

Step 8.2. Form the new target set D by setting $y_j = -1$ for all $j \in I_{\text{delete}}$.

Step 8.3. $I_{\text{remain}} \leftarrow I_{\text{remain}} \setminus I_{\text{delete}}$ and go to Step 3.

Step 9. STOP!

The given Boolean function is realized by ORing the Boolean functions corresponding to the robust templates generated by the algorithm. In the worst case, the algorithm yields templates being all minterms.

Note that in Step 6 if $\alpha = 1$, the points with $\xi_i \ge \alpha = 1$ (including the case $\xi_i = 1$ corresponding to the points on the separating hyperplane) must be included in the set I_{miss}. However, if $\alpha = 0$, the points with $\xi_i = \alpha = 0$ (corresponding to the points achieving the target functional margin of value 1) must not be included in the set I_{miss}. In Step 8.1, the indices to be deleted from I_{remain} are all indices (possibly all indices except one) in the set I_{temp}.

It is true that different values of the regularization parameter C and the desired upper bound α in Step 1 will affect the length and margins of the decomposition. In general, the choice of both parameters is problem-dependent. According to our experience, the choice of $C \ge 1$ and $\alpha = 1$ in Step 1 works well in most cases we have encountered. However, the algorithm may result in a long sequence of robust templates or poor robustness in some templates if C and α are small.

Table 2: LHJ decomposition in Example 1.

	Template values	Logic operator	Margin
1	$w = (-1,-1,-1,-1,0,-1,-1,-1,-1)$, $b = -1$		0.3536
2	$w = (-1,-1,-1,-1,-1,-1,-1,-1,-1)$, $b = -4$	XOR	0.3333

Table 3: Maxterm-based decomposition in Example 1.

	Template values	Logic operator	Margin
1	$w = (-1,-1,-1,-1,0,-1,-1,-1,-1)$, $b = -1$		0.3536
2	$w = (1,1,1,1,1,1,1,1,1)$, $b = 4$	AND	0.3333

Notice that we may also state a similar algorithm based on maxterm realization theorem, namely, the maxterm-based decomposition algorithm. In this algorithm, we simply change $y_j = -1$ to $y_j = 1$ in Steps 3, 5.1, 7.1, and 8.2 of the minterm-based decomposition algorithm.

6. Illustrative Examples

In this section, we provide six illustrative examples. In Example 1, the Boolean function is generated from the so-called Game of Life, and those in Examples 2–6 will be given by the (minimal CNN) truth tables [25], where black and white dots represent logic values 1 and −1, respectively. In the following simulations, the maxterm-based decomposition algorithm using SVC2 with $\alpha = 1$ and $C = 1$ is implemented in Examples 1–3 and the minterm-based decomposition algorithm using SVC2 with $\alpha = 1$ and $C = 2$ is implemented in Examples 4–6. We wish to point out that the template values in the following simulations are rounded to integers. All the simulated results obtained from the proposed decomposition algorithms are compared to those obtained from the LHJ algorithms with I_3.

Example 1. Consider a two-dimensional matrix of cells. The Boolean function for the Game of Life, which is not linearly separable, is generated from the following local rules.

(a) If there are two neighbors whose state values are 1, then set the state value to 1.

(b) If there are three neighbors whose state values are 1, then retain the state value.

(c) For situations apart from (a) and (b), set the state value to −1.

The LHJ and maxterm-based decompositions are shown in Tables 2 and 3, respectively. As can be seen, we obtain essentially the same results including the length and margins of the decompositions by both algorithms.

Example 2. Consider the Boolean function given by the truth table of Figure 2. The LHJ and maxterm-based decompositions are shown in Tables 4 and 5, respectively. It is seen that the maxterm-based decomposition has little shorter length but with smaller margins.

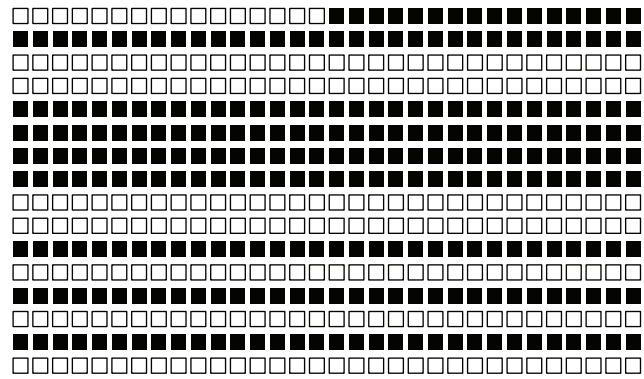

Figure 2: Boolean function of Example 2.

Table 4: LHJ decomposition in Example 2.

	Template values	Logic operator	Margin
1	$w = (-1,2,0,-1,0,0,0,0,0)$, $b = -1$		0.4083
2	$w = (-2,0,-2,1,1,0,0,0,0)$, $b = -3$	OR	0.3162
3	$w = (1,0,1,-1,0,0,0,0,0)$, $b = -2$	OR	0.5774

Table 5: Maxterm-based decomposition in Example 2.

	Template values	Logic operator	Margin
1	$w = (-3,1,1,-2,0,0,0,0,0)$, $b = 2$		0.2581
2	$w = (3,3,-2,1,1,0,0,0,0)$, $b = 5$	AND	0.2041

Example 3. Let the Boolean function be given by the truth table shown in Figure 3. The LHJ decomposition is shown in Table 6 and the maxterm-based decomposition is given in Table 7. It is observed that the length of the maxterm-based decomposition is much shorter than that of LHJ decomposition, but the minimal margin of the templates by the maxterm-based algorithm is a little bit smaller.

Example 4. Consider the Boolean function given by the truth table of Figure 4. The resulting decomposition by LHJ algorithm is given in Table 8. As shown in Table 9, the length and the minimal margin of the minterm-based decomposition are the same as those of the LHJ decomposition.

Example 5. Let the Boolean function be given by the truth table shown in Figure 5. The LHJ and minterm-based decompositions are shown in Tables 10 and 11, respectively. Note that the length of the minterm-based decomposition is shorter than that of the LHJ decomposition. Note that the first templates in both decompositions have small margins.

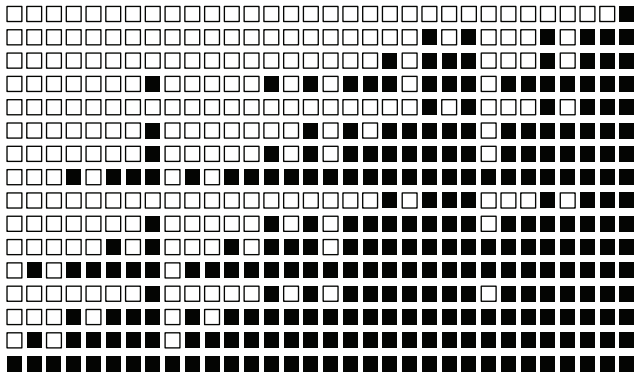

Figure 3: Boolean function of Example 3.

Table 6: LHJ decomposition in Example 3.

	Template values	Logic operator	Margin
1	$w = (3, 2, 2, 2, 3, 1, 2, 1, 2, 1)$, $b = 1$		0.1581
2	$w = (1, 1, 2, 1, 2, 0, 1, 1, 1)$, $b = 1$	AND	0.2673
3	$w = (2, 1, 0, 1, 0, 1, 1, 0, 1)$, $b = -4$	OR	0.3333
4	$w = (0, 1, 2, 1, 2, 0, 1, 1, 1)$, $b = -2$	OR	0.3333
5	$w = (1, 1, 0, 0, 0, 1, 0, 1, 1)$, $b = 4$	AND	0.4472

Table 7: Maxterm-based decomposition in Example 3.

	Template values	Logic operator	Margin
1	$w = (4, 3, 4, 3, 5, 1, 3, 2, 3)$, $b = 1$		0.1010
2	$w = (1, 1, -1, -1, -1, 1, -1, 1, 1)$, $b = 8$	AND	0.3333

Table 8: LHJ decomposition in Example 4.

	Template values	Logic operator	Margin
1	$w = (3, 1, -1, 1, 3, 2, 1, 3, 2)$, $b = 2$		0.1601
2	$w = (0, 0, 1, -1, 0, 1, 1, 1, 0)$, $b = -4$	NAND	0.4472

Table 9: Minterm-based decomposition in Example 4.

	Template values	Logic operator	Margin
1	$w = (3, 1, -1, 1, 3, 2, 1, 3, 2)$, $b = 2$		0.1601
2	$w = (-1, 0, 1, -1, -1, 1, 1, 1, -1)$, $b = -7$	OR	0.3536

Example 6. Let the Boolean function be given by the truth table shown in Figure 6. The resulting decompositions by the LHJ and minterm-based algorithms are given in Tables 12 and 13, respectively. Again, the minterm-based decomposition has shorter length but with smaller margins. In Table 13, it is seen that the first Boolean function has a smaller margin than other Boolean functions in the minterm-based decomposition.

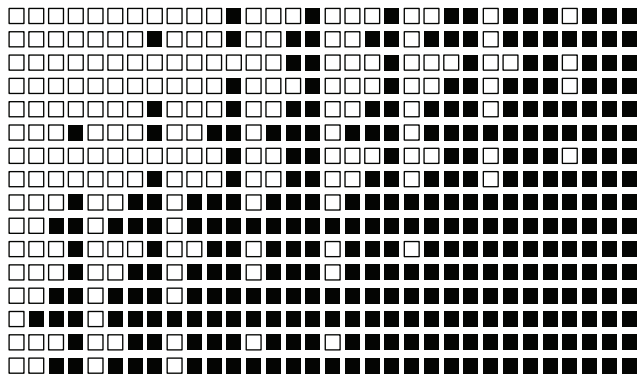

Figure 4: Boolean function of Example 4.

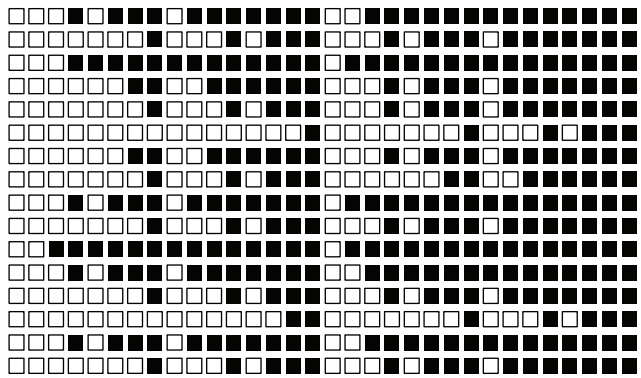

Figure 5: Boolean function of Example 5.

Table 10: LHJ decomposition in Example 5.

	Template values	Logic operator	Margin
1	$w = (1, -2, 1, -2, 2, 3, 3, 3, 2)$, $b = 2$		0.1490
2	$w = (0, -1, 1, -1, 1, 1, 1, 1, 1)$, $b = 1$	AND	0.3535
3	$w = (0, -1, 1, -1, 0, 1, 1, 0, 0)$, $b = -2$	OR	0.4472
4	$w = (1, 0, 0, 0, 0, 1, 1, 1, 0)$, $b = -3$	OR	0.5000

Table 11: Minterm-based decomposition in Example 5.

	Template values	Logic operator	Margin
1	$w = (1, -3, 2, -3, 3, 4, 4, 4, 3)$, $b = 9$		0.1060
2	$w = (-1, -1, 1, -1, -1, -1, 1, -1, -1)$, $b = -8$	OR	0.3333
3	$w = (-1, -1, 1, -1, -1, 1, -1, -1, -1)$, $b = -8$	OR	0.3333

7. Conclusion

In this study, simple minterm- and maxterm-based decomposition algorithms utilizing soft margin and maximal margin classifiers have been proposed to design a sequence of robust templates implementing an arbitrary Boolean function. In contrast to the past research works, the control template parameters and thresholds in our approach are not restricted to assume only a given finite set of integers. Several numerical examples have been provided to illustrate

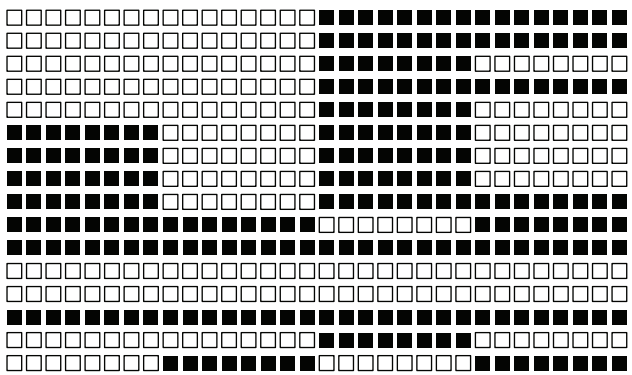

Figure 6: Boolean function of Example 6.

Table 12: LHJ Decomposition in Example 6.

	Template values	Logic operator	Margin
1	$w = (2, -2, -1, 1, 3, -2, 0, 0, 0)$, $b = 1$		0.2085
2	$w = (1, -1, 1, 2, 1, -1, 0, 0, 0)$, $b = -3$	XOR	0.3780
3	$w = (-1, 1, 0, 0, 0, -1, 0, 0, 0)$, $b = -2$	OR	0.5774
4	$w = (-1, -2, 1, 2, 1, -1, 0, 0, 0)$, $b = 3$	AND	0.2887
5	$w = (1, 1, 0, 1, 0, 1, 0, 0, 0)$, $b = -3$	OR	0.5000
6	$w = (-1, 0, 0, 0, 1, -1, 0, 0, 0)$, $b = -2$	OR	0.5774
7	$w = (1, 1, 0, 0, 1, 0, 0, 0, 0)$, $b = 2$	AND	0.5774
8	$w = (1, -1, 1, -1, 0, 0, 0, 0, 0)$, $b = -3$	OR	0.5000

Table 13: Minterm-based decomposition in Example 6.

	Template values	Logic operator	Margin
1	$w = (-5, -2, -1, 1, 5, -3, 0, 0, 0)$, $b = -8$		0.1240
2	$w = (3, 1, -3, 3, -1, 1, 0, 0, 0)$, $b = -7$	OR	0.1826
3	$w = (3, -2, 1, -3, 1, -1, 0, 0, 0)$, $b = -6$	OR	0.2000
4	$w = (-2, 2, 1, 1, -2, -2, 0, 0, 0)$, $b = -7$	OR	0.2357
5	$w = (1, 1, 1, 1, 0, 1, 0, 0, 0)$, $b = -4$	OR	0.4472

the use of the proposed method. CFC and LHJ decomposition methods are in general faster than the proposed minterm- and maxterm-based decomposition methods. However, in many cases, the minterm- and maxterm-based decomposition methods produce a sequence of robust templates with shorter length. To further speed up the decomposition algorithm, a natural idea is to make use of more logic functions as conjunctions for the sequence of robust templates, instead of OR operator in minterm-based decomposition or AND operator in maxterm-based decomposition. This constitutes an interesting future research topic. Furthermore, it is worthwhile to investigate if one can devise a decomposition algorithm such that each resulting robust template in the decomposition has a margin greater than or equal to a prespecified value. This constitutes another interesting future research topic.

Nomenclature

$\underline{p}$:	$\underline{p} := \{1, 2, \ldots, p\}$
$\mathfrak{R}^n$:	n-dimensional real space
$\langle x, x' \rangle$:	Usual Euclidean inner product of $x,\ x' \in \mathfrak{R}^n$
$\|x\|$:	Euclidean norm of $x \in \mathfrak{R}^n$
$X \times Y$:	Cartesian product of sets X and Y
$\varnothing$:	Empty set
$A \setminus B$:	Set of all elements in set A but not in B (i.e., set difference)
$\#(A)$:	Number of elements in set A
sign$(\cdot)$:	Sign function
argmax$_{i \in I}\{\xi_i\}$:	Set of all indices j in I for which ξ_j achieve the maximum
I_1:	$I_1 := \{-1, 0, 1\}$
I_2:	$I_2 := \{-2, -1, 0, 1, 2\}$
I_3:	$I_3 := \{-3, -2, -1, 0, 1, 2, 3\}$.

Acknowledgment

The research reported here was supported by the National Science Council, Taiwan, under Grant no. NSC 101-2221-E-214-073.

References

[1] L. O. Chua and L. Yang, "Cellular neural networks: theory," *Institute of Electrical and Electronics Engineers*, vol. 35, no. 10, pp. 1257–1272, 1988.

[2] L. O. Chua and L. Yang, "Cellular neural networks: applications," *Institute of Electrical and Electronics Engineers*, vol. 35, no. 10, pp. 1273–1290, 1988.

[3] Y. Kao and C. Gao, "Global exponential stability analysis for cellular neural networks with variable coefficients and delays," *Neural Computing and Applications*, vol. 17, no. 3, pp. 291–295, 2008.

[4] P. Balasubramaniam, M. S. Ali, and S. Arik, "Global asymptotic stability of stochastic fuzzy cellular neural networks with multiple time-varying delays," *Expert Systems with Applications*, vol. 37, no. 12, pp. 7737–7744, 2010.

[5] Q. Han, X. Liao, and C. Li, "Analysis of associative memories based on stability of cellular neural networks with time delay," *Neural Computing and Applications*, vol. 20, pp. 1–8, 2012.

[6] L. Wang and T. Chen, "Complete stability of cellular neural networks with unbounded time-varying delays," *Neural Networks*, vol. 36, pp. 11–17, 2012.

[7] A. Baştürk and E. Günay, "Efficient edge detection in digital images using a cellular neural network optimized by differential evolution algorithm," *Expert Systems with Applications*, vol. 36, no. 2, pp. 2645–2650, 2009.

[8] A. C. B. Delbem, L. G. Correa, and L. Zhao, "Design of associative memories using cellular neural networks," *Neurocomputing*, vol. 72, no. 10-12, pp. 2180–2188, 2009.

[9] H. Li, X. Liao, C. Li, H. Huang, and C. Li, "Edge detection of noisy images based on cellular neural networks," *Communications in Nonlinear Science and Numerical Simulation*, vol. 16, no. 9, pp. 3746–3759, 2011.

[10] F. Chen, G. He, and G. Chen, "Realization of Boolean functions via CNN: mathematical theory, LSBF and template design," *IEEE Transactions on Circuits and Systems*, vol. 53, no. 10, pp. 2203–2213, 2006.

[11] B. Mi, X. Liao, and C. Li, "Identification and realization of linearly separable Boolean functions via cellular neural networks," *International Journal of Bifurcation and Chaos*, vol. 18, no. 11, pp. 3299–3308, 2008.

[12] F. Chen, G. He, and G. Chen, "Realization of Boolean functions via CNN with von Neumann neighborhoods," *International Journal of Bifurcation and Chaos*, vol. 16, no. 5, pp. 1389–1403, 2006.

[13] M. Hänggi and G. S. Moschytz, "An exact and direct analytical method for the design of optimally robust cnn templates," *IEEE Transactions on Circuits and Systems I*, vol. 46, no. 2, pp. 304–311, 1999.

[14] G. Fornarelli and A. Giaquinto, "Adaptive particle swarm optimization for CNN associative memories design," *Neurocomputing*, vol. 72, no. 16-18, pp. 3851–3862, 2009.

[15] A. Giaquinto and G. Fornarelli, "PSO-based cloning template design for CNN associative memories," *IEEE Transactions on Neural Networks*, vol. 20, no. 11, pp. 1837–1841, 2009.

[16] M. Di Marco, M. Forti, and A. Tesi, "On the margin of complete stability for a class of cellular neural networks," *International Journal of Bifurcation and Chaos*, vol. 18, no. 5, pp. 1343–1361, 2008.

[17] S. Xavier-de-Souza, M. E. Yalçın, J. A. K. Suykens, and J. Vandewalle, "True random bit generation from a double-scroll attractor," *IEEE Transactions on Circuits and Systems I*, vol. 51, no. 5, pp. 892–902, 2004.

[18] C. T. Lin, C. L. Chang, and W. C. Cheng, "A recurrent fuzzy cellular neural network system with automatic structure and template learning," *IEEE Transactions on Circuits and Systems I*, vol. 5, no. 5, pp. 1024–1035, 2004.

[19] C. T. Lin, C. L. Chang, and J. F. Chung, "New horizon for CNN: with fuzzy paradigms for multimedia," *IEEE Circuits and Systems Magazine*, vol. 5, no. 2, pp. 20–35, 2005.

[20] C. L. Chang, K. W. Fan, I. F. Chung, and C. T. Lin, "A recurrent fuzzy coupled cellular neural network system with automatic structure and template learning," *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 8, pp. 602–606, 2006.

[21] F. Chen, G. Chen, G. He, X. Xu, and Q. He, "Universal perceptron and DNA-like learning algorithm for binary neural networks: LSBF and PBF implementations," *IEEE Transactions on Neural Networks*, vol. 20, no. 10, pp. 1645–1658, 2009.

[22] F. Chen, G. Chen, Q. He, G. He, and X. Xu, "Universal perceptron and DNA-like learning algorithm for binary neural networks: non-LSBF implementation," *IEEE Transactions on Neural Networks*, vol. 20, no. 8, pp. 1293–1301, 2009.

[23] K. R. Crounse, E. L. Fung, and L. O. Chua, "Efficient implementation of neighborhood logic for cellular automata via the cellular neural network universal machine," *IEEE Transactions on Circuits and Systems I*, vol. 44, no. 4, pp. 355–361, 1997.

[24] Y. L. Lin, J. G. Hsieh, and J.-H. Jeng, "Robust template decomposition with restricted weights for cellular neural networks implementing an arbitrary Boolean function," *International Journal of Bifurcation and Chaos*, vol. 17, no. 9, pp. 3151–3169, 2007.

[25] L. O. Chua and T. Roska, *Cellular Neural Networks and Visual Computing*, University Press, Cambridge, Cambridge, UK, 2002.

[26] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceeding of the 5th Annual Workshop on Computational Learning Theory*, pp. 144–152, New York, NY, USA, 1992.

[27] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Statistics for Engineering and Information Science, Springer, New York, NY, USA, 2nd edition, 2000.

[28] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, Cambridge, UK, 2000.