Handwritten digit recognition based on shunting inhibitory convolutional neural networks

Fok Hing Chi Tivive
University of Wollongong, tivive@uow.edu.au

Abdesselam Bouzerdoum
University of Wollongong, bouzer@uow.edu.au

Follow this and additional works at: https://ro.uow.edu.au/infopapers

Part of the Physical Sciences and Mathematics Commons

Recommended Citation
https://ro.uow.edu.au/infopapers/2861

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au
Handwritten digit recognition based on shunting inhibitory convolutional neural networks

Abstract
This paper presents the application of a new class of convolutional neural networks based on the mechanism of shunting inhibition for handwritten digit recognition. With a three layer network architecture and the use of shunting inhibitory neurons as information processing elements, the network consists of 1926 trainable parameters which are adapted by a first-order gradient training algorithm derived from Rprop, Quickprop, and SuperSAB. Trained on a dataset of 10,000 samples and evaluated on the entire test set of the MNIST database, these networks achieve classification accuracies above 93% with the best performance obtained from those networks with partial connection schemes.

Disciplines
Physical Sciences and Mathematics

Publication Details
F. Tivive & A. Bouzerdoum, "Handwritten digit recognition based on shunting inhibitory convolutional neural networks", in Workshop on Learning Algorithms for Pattern Recognition in Conjunction with the 18th Australian Joint Conference on Artificial Intelligence (AI'05), 2005, pp. 72-77.

This conference paper is available at Research Online: https://ro.uow.edu.au/infopapers/2861
HANDWRITTEN DIGIT RECOGNITION BASED ON SHUNTING INHIBITORY
CONVOLUTIONAL NEURAL NETWORKS

Fok Hing Chi Tivive and Abdesselam Bouzerdoum, Senior Member, IEEE
School of Electrical, Computer and Telecommunication Engineering
University of Wollongong
Northfields Avenue, Wollongong, NSW 2522, Australia.
E-mail:[fhct243@uow.edu.au], [a.bouzerdoum@elec.uow.edu.au]

ABSTRACT

This paper presents the application of a new class of convolutional neural networks based on the mechanism of shunting inhibition for handwritten digit recognition. With a three layer network architecture and the use of shunting inhibitory neurons as information processing elements, the network consists of 1926 trainable parameters which are adapted by a first-order gradient training algorithm derived from Rprop, Quickprop, and SuperSAB. Trained on a dataset of 10,000 samples and evaluated on the entire test set of the MNIST database, these networks achieve classification accuracies above 93% with the best performance obtained from those networks with partial-connection schemes.

1. INTRODUCTION

After the derivation of error backpropagation algorithm as a training method for multilayer perceptrons (MLPs) by Rumelhart et al. [1], MLPs have been applied intensively to a broad spectrum of applications, in areas as diverse as finance, medicine, engineering, geology, and physics; indeed, in any problem that consists of prediction, classification or control. However, MLPs suffer from several drawbacks when they are operated directly on “raw” input images where large amounts of data are available. One of the drawbacks is the number of network parameters increases with respect to the dimension of the input. Therefore, in the past two decades, researchers have not only focused on the development of training algorithms for MLPs but also on the identification of significant network structure and weight constraints that can reduce the number of free parameters to be trained, and consequently the complexity of training. They have extended the feedforward network architecture by arranging the neurons in the hidden layers into planes and applying some biological concepts to develop two-dimensional (2-D) network structures that can incorporate prior knowledge about the task into the networks. This type of structured networks is known as convolutional neural networks (CoNNs) and has gained considerable interests in solving visual pattern recognition tasks. In the literature, there are several types of CoNNs that have been proposed; however, most of them were tailored for specific applications [2-4].

In [5], we proposed a new class of convolutional neural networks based on the physiologically plausible mechanism of shunting inhibition, known as shunting inhibitory convolutional neural network, or SICoNNet for short. These networks have a flexible architecture in which the user only specifies the input size, the receptive field size, the number of layers and/or number of feature maps, the number of outputs, and the connection scheme between layers. The processing elements in the hidden layers of the network are shunting inhibitory neurons, whereas in the output layer, the output units are sigmoid neurons. This type of CoNNs has been used in a face detection and localization system that can discriminate face patterns from complex background scenes with a high detection accuracy [6]. In this paper, we have extended the network architecture so that it can be applied to multi-class pattern recognition problems such as handwritten digit recognition.

The remainder of the paper is organized as follows. The next section describes the architecture of SICoNNet which has been modified so that with partial-connection schemes, the number of feature maps in the first hidden layer of the network can be arbitrary changed. Section 3 describes a batch training method that has been developed for training the proposed CoNNs. Section 4 presents the experimental results and performance analysis of the networks. Finally, concluding remarks are presented in Section 5.

2. DESCRIPTION OF THE NETWORK MODEL

Compared with some existing CoNNs [4, 7, 8] in which the connection strategy is not trivial and manually chosen, SICoNNets were developed relying on three systematic connection schemes, namely binary-connection, toepplitz-connection, and full-connection. In the full-connection scheme, each hidden layer of the network has an arbitrary...
number of feature maps which are fully connected to those feature maps in the succeeding layer. This scheme is similar to MLP, where the number of hidden layer and hidden neurons (equivalent to feature maps) can be varied. In a binary connection scheme, each feature map branches out to two feature maps in the successive layer similar to a binary tree, whereas in the toeplitz-connection scheme, a feature map may have one-to-one or one-to-many links with feature maps in the preceding layer. As an example, Table I illustrates the connections between Layer 2 (L-2) and Layer 3 (L-3). Suppose that Layer 3 contains eight feature maps labeled 1 to 8 (first column), and Layer-2 has four feature maps labeled A, B, C and D. Feature maps 1 and 8 have one-to-one connections with feature maps A and D, respectively. Feature map 2 has connections with feature maps B and A; the rest of the connections form a toeplitz matrix. In other words, each feature map in Layer 2 connects to the same number of feature maps in Layer 3 (in this case five), and its connections appear along a diagonal of the connection matrix.

Table I Connections between feature maps in layer-2 and layer-3 of the toeplitz-connection scheme.

<table>
<thead>
<tr>
<th>Layer-3 Feature maps</th>
<th>Connections from L-2 to L-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>B A</td>
</tr>
<tr>
<td>3</td>
<td>C B A</td>
</tr>
<tr>
<td>4</td>
<td>D C B A</td>
</tr>
<tr>
<td>5</td>
<td>D C B A</td>
</tr>
<tr>
<td>6</td>
<td>D C B</td>
</tr>
<tr>
<td>7</td>
<td>D C</td>
</tr>
<tr>
<td>8</td>
<td>D</td>
</tr>
</tbody>
</table>

In both partial-connection schemes, the number of feature maps in each hidden layer is constrained to an integer power of 2, e.g., the number of feature maps in the Lth hidden layer is equal to $2^L$. However, this constraint prevents the network from extracting various types of features in the input image as the first hidden layer of the partially-connected networks can only have two feature maps. To alleviate this drawback, we have slightly modified the network structure for the partial-connection schemes so that the number of feature maps in the first hidden layer can be changed arbitrarily. For instance, in the binary-connected network, the number of feature maps in the first hidden layer can be altered so long as the successive layer has twice the number of feature maps, and each feature map is connected to two feature maps in the following layer. For the toeplitz-connection scheme, the first hidden layer can have $2^K$ feature maps, and the successive layer has twice that number of planes, where the value of $K$ is chosen arbitrarily. The advantages of using a partial-connection scheme are

- firstly it reduces the number of connections within the network, which may increase the generalization ability;
- secondly it diversifies the extraction of high-order features by taking inputs from different sets of feature maps in the previous layer.

For this pattern recognition problem, the network architecture is a three layer network with four feature maps in the first hidden layer and eight feature maps in the second hidden layer. The input layer is a 2-D array of size 28x28, and the output layer consists of ten sigmoid neurons to represent the ten numerical characters. The feature map is made up of a lattice of shunting inhibitory neurons which receive inputs from a local neighborhood called the receptive field in the input image. The size of the receptive field used throughout the network is 5x5. The activation of a shunting inhibitory neuron at location $(i, j)$ in feature map $(L, k)$ (kth feature map of the Lth layer) can be mathematically described by

$$Z_{L,k}(i,j) = g_L \left[ \sum_{v=1}^{S_L} [C_{L,k,v} \ast Z_{L-1,v}(i,j)] + b_{L,k}(i,j) \right] + a_{L,k}(i,j) \right) + d_{L,k}(i,j)$$

where $Z_{L-1,v}$ represents the output plane of the vth feature map of the $(L-1)$th layer, $S_{L-1}$ denotes the number of feature maps in the $(L-1)$th layer, $C_{L,k}$ and $D_{L,k}$ are the set of weights (convolution masks), $a_{L,k}(i,j)$ is the passive decay rate, $b_{L,k}(i,j)$ and $d_{L,k}(i,j)$ are bias terms, and the “*” is the 2-D convolution operator. In the first layer, the activation functions $g_L$ and $f_L$ are chosen to be the hyperbolic tangent and exponential functions, respectively, whereas in the second layer, $g_L$ is set to the logarithmic sigmoid function. To minimize the number of network parameters, each feature map has only one set of weights, which is shared among all neurons; this mechanism is known as weight sharing. Furthermore, the bias parameters and the passive decay rate are shared by all neurons in the feature map. In other words, a feature map has one shunting neuron which is replicated into a 2-D array. A sub-sampling operation is performed within each hidden layer to decrease the size of the feature maps by one quarter. This is done by shifting the centers of receptive fields of neighboring units by two positions, horizontally and vertically. Through this downsampling process, some information about the exact location of the detected feature is discarded; once a feature is detected its absolute location is no longer important, only its position relative to other features is relevant to the
classification [12]. This introduces some degree of shift and distortion tolerance into the network. At the last hidden layer, a local average operation is applied on all the feature maps; i.e., small 2x2 non-overlapping regions are averaged and the resulting signals are fed into the output layer. As the size of the feature map in the last hidden layer is 7x7, the feature maps are padded with zero before the local average operation is performed. Figure 1 shows the modified binary-connected network architecture.

Quickprop [10], and SuperSAB [11] has been developed. The weight update rule is expressed as

$$ \tilde{W}(k + 1) = \tilde{W}(k) + \Delta \tilde{W}(k) + \bar{\mu}(k) \cdot * \Delta \tilde{W}(k - 1), $$

where "*" is the element-by-element product of two column vectors. The weight update $\Delta \tilde{W}(k)$ is computed using the same principle as the Rprop algorithm, i.e., each local weight, $w_i(k)$, in the weight vector $\tilde{W}(k)$ has its own step size, $\gamma_i(k)$, which is adjusted according to the observation of the behavior of the local gradient, $g_i(k)$, during two successive iterations

$$ \gamma_i(k) = \begin{cases} \min(1.2 \gamma_i(k - 1), \gamma_{\max}), & \text{if } g_i(k) g_i(k - 1) > 0 \\ \min(0.5 \gamma_i(k - 1), \gamma_{\min}), & \text{if } g_i(k) g_i(k - 1) < 0 \\ \gamma_i(k - 1), & \text{otherwise} \end{cases} $$

where $n$ is the number of trainable weights, $\gamma_{\max}$ and $\gamma_{\min}$ are the upper and lower limits of the step size, respectively, and they are set accordingly: $\gamma_{\max} = 10$ and $\gamma_{\min} = 10^{-10}$; the initial value $\gamma_i(0)$ is set to 0.001. Then, the local weight update of the $i$th weight is determined by

$$ \Delta w_i(k) = -\text{sgn}(g_i(k)) \gamma_i(k), $$

where "sgn" is the signum function. When the current local gradient has a change of sign with respect to the previous local gradient of the same weight, the stored local gradient is set to zero so as to avoid an update in that weight in the next iteration. Furthermore, when the product of the current and previous local gradients is less than zero together with an increase in the network error, $E$, the $i$th weight update is reverted to the previous weight update multiplied by an adaptive momentum rate

$$ \text{if } g_i(k) g_i(k - 1) < 0 \text{ and } E(k) > E(k - 1) \text{ then } \Delta w_i(k) = -\mu_i(k) \Delta w_i(k - 1). $$

The adaptive momentum rate $\mu_i(k)$ is calculated by taking the magnitude of the Quickprop-step and is calculated by

$$ \tilde{\mu}_i(k) = \frac{|g_i(k)|}{|g_i(k - 1) - g_i(k)|}, $$

and is bounded as follows:

$$ \mu_i(k) = \begin{cases} \max(\tilde{\mu}_i(k), 0.5), & \text{if } g_i(k) g_i(k - 1) < 0 \\ 0, & \text{if } g_i(k) g_i(k - 1) = 0 \end{cases} $$

3. TRAINING ALGORITHM

To train the aforementioned networks, a batch training algorithm based on the combination of Rprop [9],
Hence, the first-order gradient training method can be summarized by the following pseudo-code:

\[
\mu(k) = \min(\bar{\mu}(k), 1.5) .
\]

In addition, when there is a decrease in the current network error with respect to the previous error, a small percentage of the negative gradient is added to the weight

\[
\tilde{W}(k + 1) = W(k + 1) - \alpha(k) \cdot \tilde{g}(k)
\]

where \( \alpha(k) \) is the vector of learning rates and is adapted using similar principle as the SuperSAB method

\[
\alpha_i(k) = \begin{cases} 
1.2\alpha_i(k - 1), & \text{if } g_i(k)g_i(k - 1) > 0 \\
0.5\alpha_i(k - 1), & \text{if } g_i(k)g_i(k - 1) < 0 \\
\alpha_i(k - 1), & \text{otherwise}
\end{cases} 
\]

To prevent the learning rate from increasing indefinitely, it is bounded above by the following expression

\[
\alpha_i(k) = \min(\bar{\alpha}_i(k), 0.9) .
\]

Hence, the first-order gradient training method can be summarized by the following pseudo-code:

```
Input: Initialize \( \alpha_i \leftarrow 0.001, \mu_i \leftarrow 0.01, \alpha_i \leftarrow 0.1 \). Calculate the local gradient.
1: while stopping criterion is not met do
2: Calculate the adaptive momentum rate \( \bar{\mu}_i(k) \), according to (7).
3: if \( g_i(k)g_i(k - 1) > 0 \) then
4: \( \gamma_i(k) \leftarrow \min(1.2\gamma_i(k - 1), \gamma_{\max}) \),
5: \( \bar{\alpha}_i(k) \leftarrow 1.2\alpha_i(k - 1) \).
6: else if \( g_i(k)g_i(k - 1) < 0 \) then
7: \( \gamma_i(k) \leftarrow \max(0.5\gamma_i(k - 1), \gamma_{\min}) \),
8: \( \bar{\alpha}_i(k) \leftarrow 0.5\alpha_i(k - 1) \),
9: \( \mu_i(k) \leftarrow \max(\bar{\mu}_i(k), 0.5) \),
10: \( g_i(k) \leftarrow 0 \).
11: else if \( g_i(k)g_i(k - 1) = 0 \) then
12: \( \mu_i(k) \leftarrow 0 \),
13: \( \gamma_i(k) \leftarrow \gamma_i(k - 1) \),
14: \( \bar{\alpha}_i(k) \leftarrow \alpha_i(k - 1) \).
15: end if
16: \( \alpha_i(k) \leftarrow \min(\bar{\alpha}_i(k), 0.9) \).
17: \( \Delta w_i(k) \leftarrow -sgn(g_i(k))\gamma_i(k) \).
18: if \( g_i(k)g_i(k - 1) < 0 \) and \( E'(k) > E'(k - 1) \) then
19: \( \Delta w_i(k) \leftarrow -\mu_i(k)\Delta w_i(k - 1) \).
20: end if
21: \( w_i(k + 1) \leftarrow w_i(k) + \Delta w_i(k) + \mu_i(k)\Delta w_i(k - 1) \).
22: if \( E'(k) < E'(k - 1) \) then
23: \( w_i(k + 1) \leftarrow w_i(k + 1) - \alpha_i(k)g_i(k) \).
24: end if
25: end while
```

4. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

To test all three connection schemes, three networks were developed which have the same number of trainable parameters, i.e., 1926 weights, but with different number of connections. The network parameters are initialized with random values using a uniform distribution. The weights of the receptive fields are initialized within the range \([-1/t, 1/t]\), where \( t \) is the width of the receptive fields. The bias parameters in the feature maps are initialized similarly with \( t = 1 \), and the passive decay rate term varies in the range \([0, 1]\). The following constraint is imposed on the passive decay rate parameter

\[
a_{d,k}(t, j) + f_L \left( \sum_{l \in L_{d,k}} [Z_{L_{d,k}}(j) + d_{d,k}(t, j)] \right) \geq c
\]

so as to avoid division by zero in (1); this constraint is maintained throughout the training process (here \( c = 0.1 \)). The input data used for training the network was obtained from the MNIST database [12], where a training set of 10,000 patterns with 1000 examples per digit class was generated. Each digit pattern is labeled with a vector of ten binary numbers, e.g., an image pattern of digit zero is represented by the following column vector \([1 0 0 0 0 0 0 0 0 0]\). The networks were evaluated on the entire test set of the MNIST database. As each sigmoid neuron at the output layer produces a network score, the neuron with the maximum network score is assumed to be the correct class. To have a comprehensive analysis of the trained networks, a confusion matrix together with the classification rate of each actual class is tabulated for each connection scheme.

Tables II, III, and IV show the experimental results as confusion matrices for binary-, toeplitz- and fully-connected networks, respectively, based on a single experiment.

From these confusion matrices, it is clear that the proposed networks with multiple outputs can achieve high classification performances. All three networks can recognize all ten digits with correct classification rate above 93%. The best network is the binary-connected network with classification accuracy of 94.71%, followed by the toeplitz-connected network with 94.66% and 93.38% for the fully-connected network. In general, the partially-connected networks outperform the fully-connected network, which supports the claim made in Section 2. One of the reasons why the fully-connected network performs poorly compared to the other two schemes is that each feature map has only one pair of receptive fields to extract excitatory and inhibitory inputs from the input image. Therefore, by taking inputs
Table II Confusion matrix for binary-connected network tested on 10,000 handwritten digits.

<table>
<thead>
<tr>
<th>Actual class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>954</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>8</td>
<td>2</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1106</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>0</td>
<td>991</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>7</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>13</td>
<td>958</td>
<td>1</td>
<td>15</td>
<td>0</td>
<td>7</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>923</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>13</td>
<td>13</td>
<td>3</td>
<td>857</td>
<td>3</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>9</td>
<td>10</td>
<td>919</td>
<td>0</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>14</td>
<td>12</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>950</td>
<td>2</td>
<td>39</td>
</tr>
<tr>
<td>8</td>
<td>17</td>
<td>2</td>
<td>7</td>
<td>12</td>
<td>9</td>
<td>6</td>
<td>11</td>
<td>8</td>
<td>887</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>18</td>
<td>12</td>
<td>11</td>
<td>0</td>
<td>15</td>
<td>13</td>
<td>926</td>
</tr>
</tbody>
</table>

Total classification accuracy 97.35

Table II Confusion matrix for toeplitz-connected network tested on 10,000 handwritten digits.

<table>
<thead>
<tr>
<th>Actual class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>941</td>
<td>2</td>
<td>11</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1112</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>1</td>
<td>981</td>
<td>7</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>12</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>0</td>
<td>14</td>
<td>946</td>
<td>0</td>
<td>9</td>
<td>3</td>
<td>8</td>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>927</td>
<td>1</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>31</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>1</td>
<td>9</td>
<td>17</td>
<td>1</td>
<td>831</td>
<td>8</td>
<td>0</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>923</td>
<td>0</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>5</td>
<td>18</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>964</td>
<td>3</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>12</td>
<td>2</td>
<td>8</td>
<td>20</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>902</td>
<td>11</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>15</td>
<td>13</td>
<td>11</td>
<td>1</td>
<td>12</td>
<td>6</td>
<td>939</td>
</tr>
</tbody>
</table>

Total classification accuracy 96.60

Table II Confusion matrix for fully-connected network tested on 10,000 handwritten digits.

<table>
<thead>
<tr>
<th>Actual class</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>953</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1107</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>1</td>
<td>972</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>9</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>13</td>
<td>939</td>
<td>1</td>
<td>19</td>
<td>2</td>
<td>15</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td>913</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>34</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>20</td>
<td>4</td>
<td>833</td>
<td>11</td>
<td>2</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>4</td>
<td>7</td>
<td>0</td>
<td>6</td>
<td>11</td>
<td>915</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>5</td>
<td>16</td>
<td>12</td>
<td>16</td>
<td>3</td>
<td>2</td>
<td>919</td>
<td>6</td>
<td>48</td>
</tr>
<tr>
<td>8</td>
<td>23</td>
<td>5</td>
<td>12</td>
<td>13</td>
<td>7</td>
<td>10</td>
<td>5</td>
<td>12</td>
<td>873</td>
<td>14</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>5</td>
<td>4</td>
<td>11</td>
<td>21</td>
<td>8</td>
<td>0</td>
<td>26</td>
<td>13</td>
<td>914</td>
</tr>
</tbody>
</table>

Total classification accuracy 93.38

From all the feature maps in the previous layer, the network has difficulties to train its receptive fields to extract a set of consistent features. To circumvent this problem, the feature map may have several pairs of receptive fields to connect to different feature maps, but this generates a large number of weights, which in turns reduces the network generalization ability. In contrast, the partial-connection schemes constrain the number of connections between layers, which facilitates the training of the receptive fields into feature detectors; that is why the binary-connected network has the highest classification performance. Based on the above results, our networks are not among the best existing handwritten digit recognition systems. To achieve the state-of-the-art performance, Simard et al. [13] trained their neural-based handwritten digit system on a very large number of training patterns which were pre-processed to incorporate various distortions that may occur in the test set. In [14], Calderón et al. used a boosting technique to combine several CoNNs.
so as to improve the performance of their optical character recognition system. Other systems have used a voting or rejection-based algorithm for classification. Most of the existing systems are based on networks that have several layers of neurons with thousands of trainable weights, e.g., the LeNet-5 [12] which has 60,000 trainable parameters. In contrast, the proposed networks have a compact network structure with two hidden layers and twelve feature maps. Moreover, the proposed CoNNs were trained with a batch training algorithm for a short period of time (500 epochs), whereas most of the existing systems used an online learning rule for training.

5. CONCLUSION

We have presented a new type of neural networks that can be used for handwritten digit recognition. Tested on the MNIST database, the network with a binary-connection scheme achieved the highest performance with a classification accuracy of 94.71% when trained on a small training set of 10,000 patterns. Although, the generalization error rate is greater than the existing neural-based systems, our convolutional neural network has a much simpler architecture with three layers (excluding the input layer) and eight planes acting as feature detectors. Totally, the network has 1926 trainable parameters which were adapted by a batch training method based on existing first-order gradient algorithms for 500 epochs. The performance of these networks can be improved by training with pre-processed patterns that have a wide range of affine transformations. Furthermore, instead of taking the output neuron with maximum network score as the correct class, a voting scheme can be used to determine which output neuron is the correct class.

6. REFERENCES


