

# Genetic-algorithm-based Local Binary Convolutional Neural Network for Gender Recognition

Chun-Hui Lin,<sup>1</sup> Cheng-Jian Lin,<sup>2,3\*</sup> and Shyh-Hau Wang<sup>1</sup>

<sup>1</sup>Department of Computer Science and Information Engineering, National Cheng Kung University,  
Tainan City 701, Taiwan, ROC

<sup>2</sup>Department of Computer Science and Information Engineering, National Chin-Yi University of Technology,  
Taichung City 411, Taiwan, ROC

<sup>3</sup>College of Intelligence, National Taichung University of Science and Technology,  
Taichung City 404, Taiwan, ROC

(Received December 29, 2020; accepted March 26, 2021)

**Keywords:** convolutional neural network, local binary convolution, genetic algorithm, gender classification

At present, the main focus in the development of convolutional neural networks (CNNs) is deepening the network model to improve accuracy. However, this may increase the numbers of parameters and calculations in the network architecture. When the network model is applied to mobile devices and embedded systems, the storage capacity, computing performance, and memory will become major limitations. A local binary convolutional neural network (LBCNN) has been proposed to reduce the numbers of parameters and calculations. In the LBCNN, the convolutional layer of the CNN is replaced by a local binary convolution (LBC) module. In the LBC module, there is a pre-initialized fixed parametric filter layer. Since the parameters of the filter are generated in a random manner, the result is different each time and therefore unstable. Therefore, to provide a stable and efficient recognition technique for image sensors, we propose a genetic-algorithm-based local binary convolutional neural network (GA-LBCNN) for gender recognition in this study. The genetic algorithm (GA) is used to search for the best filter parameters of the LBCNN. LeNet is adopted as the basic model architecture, and two datasets acquired from image sensors, the CIA and MORPH datasets, are used to perform face gender classification. According to the evaluation results, LBC successfully reduces the numbers of parameters and calculations. Experimental results show that the classification accuracy of the proposed GA-LBCNN reaches 88.8 and 98.2% for the CIA and MORPH datasets, respectively. Compared with the conventional LBCNN, the classification accuracy of the proposed GA-LBCNN is increased by 7.2 and 1.1%, respectively, for the two datasets.

## 1. Introduction

Owing to the substantial increases in the power of computer hardware equipment and computing performance, artificial intelligence applications have flourished. Since LeCun first

---

\*Corresponding author: e-mail: cjlin@ncut.edu.tw  
<https://doi.org/10.18494/SAM.2021.3268>

proposed LeNet,<sup>(1)</sup> many convolutional neural network (CNN) models have appeared one after another, such as AlexNet,<sup>(2)</sup> VGGNet,<sup>(3)</sup> GoogLeNet,<sup>(4)</sup> ResNet, and DenseNet.<sup>(5,6)</sup> Both AlexNet and VGGNet improve accuracy by increasing the number of layers. As a result, the network model size and the numbers of parameters and floating point operations (FLOPs) have increased significantly. As a result, CNNs can only be used with high-performance equipment. To resolve these problems, we improve the standard CNN model and reduce the numbers of parameters and calculations to achieve higher accuracy in this study. The improved CNN can run on mobile devices, such as devices with limited computing performance, storage space, and memory size. This allows real-time processing of the data acquired from sensors such as those in autonomous cars and smart cameras.

Many studies have pointed out that model quantization can effectively reduce the model size, storage space, and memory size, making models easier to apply to portable devices. For instance, Rastegari *et al.*<sup>(7)</sup> quantized the parameters in a network to approximate convolutions using binary operations. This resulted in 58 times faster convolutional operations and 32-fold less memory use. Zhou *et al.*<sup>(8)</sup> proposed a scale estimation quantization approach by analyzing the error variance acquired by the quantization process to avoid significant accuracy degradation. Furthermore, backward approximation was applied to manage the gradient mismatch problem in backward propagation. They concluded that both the compression and acceleration abilities are guaranteed by utilizing intermediate integers in quantization; moreover, the method reaches state-of-the-art performance and can be flexibly used on various networks and with different datasets. Jacob *et al.*<sup>(9)</sup> proposed a quantization program that allows inference to be executed using integer-only arithmetic, which can be implemented more efficiently than floating point inference on commonly accessible integer-only hardware. As a result, the proposed approach improved the tradeoff between on-device latency and accuracy. Han *et al.*<sup>(10)</sup> introduced a deep compression method including pruning, Huffman coding, and trained quantization. Pruning removes redundant weight links and quantization reduces the number of bits that represent each connection. In other words, some scholars have modified the architecture of CNNs. Howard *et al.*<sup>(11)</sup> used depthwise separable convolutions to build lightweight deep neural networks. Ou and Li<sup>(12)</sup> proposed vector kernels of size  $k \times 1$  or  $1 \times k$  for each convolutional layer. Iandola *et al.*<sup>(13)</sup> proposed SqueezeNet, which replaces  $3 \times 3$  filters with  $1 \times 1$  filters and reduces the number of input channels to  $3 \times 3$  filters. These strategies are desirable to decrease the number of parameters in a CNN while attempting to maintain accuracy. Juefei-Xu *et al.*<sup>(14)</sup> proposed local binary convolutional neural networks (LBCNNs), which reduce the numbers of parameters and calculations by replacing the general convolutional layer with a local binary convolution (LBC) layer. Almowallad and Sanchez<sup>(15)</sup> used an emotion distribution learning (EDL)-LBCNN framework for distribution learning of human emotions. The EDL-LBCNN incorporates an LBC layer into a CNN in order to enhance the feature extraction ability. The framework contains two streams: a four-layer CNN and a single LBC layer. The feature maps extracted by the two streams are concatenated and utilized as inputs to fully connected layers. However, the convolution operation in the LBC layer uses a randomly generated filter, which means that the results obtained are different every time and are unstable. Therefore, finding the best mask parameters is a problem to be solved.

Evolutionary computing has been successful in solving engineering tasks ranging from the molecular to the astronomical. In practical applications, the performance of CNNs is highly dependent on their parameters, so many evolutionary computing methods are employed in designing neural network architectures or even selecting better parameters of CNNs. Suganuma *et al.*<sup>(16)</sup> attempted to construct CNN architectures automatically using genetic programming. The Cartesian genetic programming (CGP) encoding approach has been applied to represent the CNN architecture and connectivity. Furthermore, comparatively highly functional modules were adopted as the node functions in CGP to narrow the search space. The experimental results showed that the CGP encoding approach can automatically find a CNN architecture with competitive performance to that of architectures obtained with state-of-the-art models. Baker *et al.*<sup>(17)</sup> introduced MetaQNN, a meta-modeling algorithm based on reinforcement learning, which is able to produce high-performing CNN architectures for a given learning task. The experimental results showed that MetaQNN can be employed in different problem settings, including supervised and unsupervised settings. Sinha *et al.*<sup>(18)</sup> proposed an approach incorporating particle swarm optimization to select the optimal image size, number of filters, filter size, and number of CNN layers. Wang *et al.*<sup>(19)</sup> proposed a hybrid differential evolution CNN, which adopts an IP-based encoding strategy to encode attributes of CNN layers, and new mutation and crossover operators were developed for variable-length CNN architectures. Ma *et al.*<sup>(20)</sup> used the genetic algorithm (GA) to find the optimal layer combination of CNN architectures for solving classification problems. The GA finds the best chromosome solution by processing the gene chromosome selection, crossover, and mutation. It has the characteristics of a group search and only uses fitness functions for the evaluation. Therefore, they chose the GA to select the filter parameters in the LBC layer.

In this study, an efficient GA-LBCNN is proposed for gender recognition. In the GA-LBCNN, the GA is used to find the best filter parameters of the LBCNN. The LeNet network is adopted in this study. The major contributions of this study are as follows:

1. The proposed GA-LBCNN solves the problem of fixed filter parameters in the LBC layer to improve accuracy.
2. Compared with the conventional LeNet, the proposed GA-LBCNN has fewer parameters and calculations in the convolution part.
3. In tests using the CIA and MORPH datasets, it is found that the accuracy of the proposed GA-LBCNN reaches 88.8 and 98.2%, respectively, which are higher than the values obtained with the conventional LeNet.

The rest of this paper is organized as follows. Section 2 introduces the proposed GA-LBCNN structure. The experimental results obtained using two facial datasets are described in Sect. 3. Section 4 gives conclusions.

## 2. Proposed GA-LBCNN

This section introduces the use of the GA to optimize the local-binary-based LeNet (LB-LeNet) parameters. In Sect. 2.1, we explain the difference between the conventional LeNet and LB-LeNet. In Sect. 2.2, the overall architecture of the GA-LBCNN after adding the GA is described.

## 2.1 Local-binary-based LeNet (LB-LeNet)

To reduce the number of parameters in the model, we replaced the conventional convolutional layer in LeNet with the LBC layer in the LBCNN. Figure 1 illustrates the LBC layer. The binary parameters in the LBC layer can effectively reduce the memory space. The architecture diagrams of the LBC layer and the conventional convolutional layer are respectively shown in Figs. 2(a) and 2(b). Figure 2(a) displays the network architecture of LeNet. First, the input image is processed using the convolutional layer to obtain the feature map. The value is adjusted for nonlinear changes using the activation function. Then, the feature is concentrated using the pooling layer to reduce the size of the feature and the number of calculations. Finally, the calculation of the fully connected layer is used to obtain the confidence level of each category and turn it into the output result. In Fig. 2(b), LB-LeNet replaces the green  $5 \times 5$  convolutional layer in Fig. 2(a) with the gray  $5 \times 5$  fixed binarization parameters and the green  $1 \times 1$  convolutional layer in Fig. 2(b). Since the binary convolutional filters in LB-LeNet have fixed parameters, we use the GA to obtain the optimized parameters of the gray part of Fig. 2(b).

## 2.2 Proposed GA-LB-LeNet

In this study, we adopt the LeNet network as a CNN network. Therefore, the proposed GA-LBCNN is also called GA-LB-LeNet in this paper. The GA is a model derived from biological evolution operations including selection, crossover, and mutation. The concept is that better

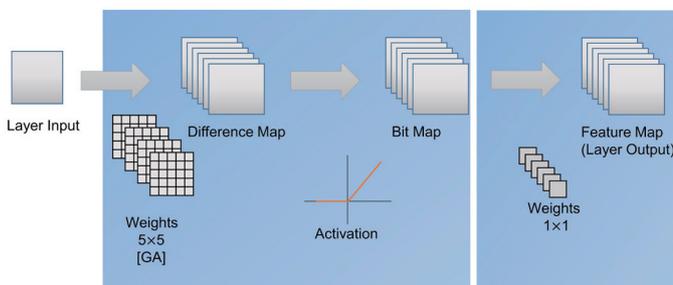


Fig. 1. (Color online) LBC layer.

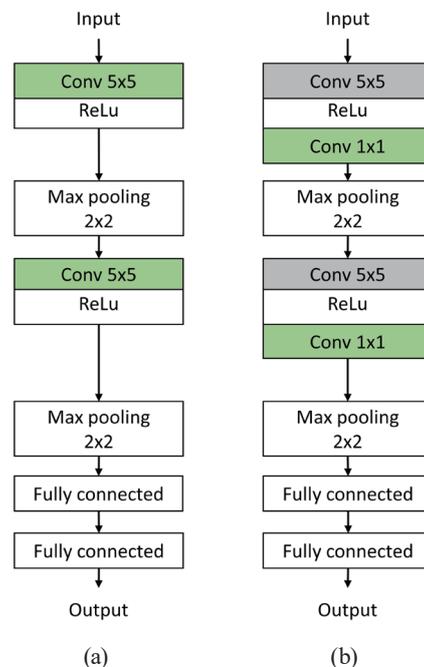


Fig. 2. (Color online) (a) Diagram of LeNet architecture and (b) diagram of LB-LeNet architecture.

biological genes can be passed to the next generation, and the best solution can be found after multiple generations of evolution through mathematical calculations. Figure 3 illustrates the coding method of chromosomes in the GA. It converts the binary weighted convolutional layer of each layer into a one-dimensional code. The coding length of each chromosome is given by Eq. (1). For example, if the input is three channels and the output is a  $5 \times 5$  convolution kernel with eight channels, then a chromosome will be converted to one-dimensional information with a length of 600.

$$Length = \sum_{L=1}^N (K^2 * C_{in} * C_{out}) \tag{1}$$

Here,  $N$  is the number of LBC layers,  $K$  is the filter size,  $C_{in}$  is the number of input channels, and  $C_{out}$  is the number of output channels.

The process of optimizing the LB-LeNet parameters using the GA algorithm is shown in Fig. 4. The steps of the proposed GA-LB-LeNet are as follows:

- Step 1: Initialize chromosome population individuals. Each individual represents the parameters of a  $5 \times 5$  filter, where these parameters are binary parameters.
- Step 2: Update the parameters of the  $1 \times 1$  convolutional layer and the fully connected layer in the network through the backpropagation algorithm.
- Step 3: Calculate the fitness values of the network according to the updated parameters, where the fitness value is the accuracy of the test data.
- Step 4: Determine whether the termination condition is met. If the maximum number of generations is reached, the best individual is obtained and the program is terminated: if it is not reached, Step 5 is executed.
- Step 5 Retain the chromosome with the highest fitness value, then take out  $(n-1)$  chromosomes from the population according to roulette wheel selection. The probability of each chromosome in the roulette wheel selection is as follows:

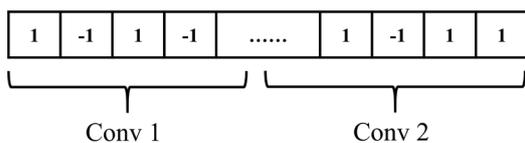


Fig. 3. Coding method of chromosomes in GA.

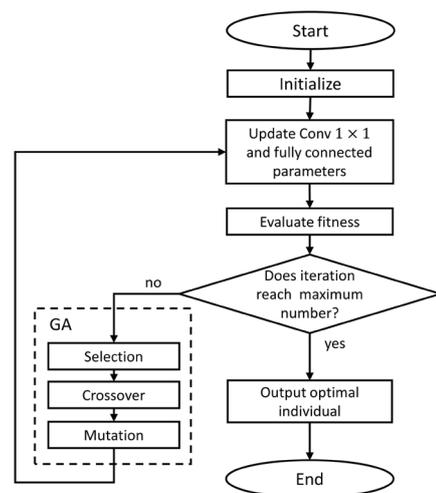


Fig. 4. Proposed GA-LB-LeNet.

$$P(x_i) = \frac{Fitness(x_i)}{\sum_{j=1}^k Fitness(x_j)}, \quad (2)$$

where  $k$  is the number of chromosomes.

Step 6: The chromosomes selected according to Step 5 produce  $n$  new chromosomes through the crossover operation.

Step 7: The chromosomes selected according to Step 5 generate  $n$  new chromosomes through the mutation operation.

Step 8: Use the chromosomes generated by Steps 5–7 as new chromosomes, and then return to Step 2 to continue to perform these steps in sequence.

### 3. Experimental Results

In this section, to verify the effectiveness of the proposed method, two datasets acquired from image sensors, the CIA and MORPH datasets, are used for testing. In this section, the two datasets are first introduced. Then, they are compared, with a discussion of the accuracy and numbers of parameters and calculations of the proposed GA-LB-LeNet, LB-LeNet, and the conventional LeNet.

#### 3.1 CIA and MORPH datasets

The CIA dataset<sup>(21)</sup> is a small Chinese facial image dataset, as shown in Fig. 5. The total number of images is 2088 (1080 male and 1088 female), and the age distribution ranges from 6 to 80 years old. This dataset contains face images with different environments, light sources, and expressions.

The MORPH dataset<sup>(21)</sup> is a multiracial face dataset, as shown in Fig. 6. The total number of images is 55134 (46645 male and 8489 female). The dataset contains face images of various races including Africa, Europe, Asian, and Hispanic. The age distribution ranges from 16 to 77 years old.



Fig. 5. (Color online) CIA dataset.



Fig. 6. (Color online) MORPH dataset.

### 3.2 Experimental and analysis results

This study uses accuracy as the standard for evaluating the system, and its formula is as follows:

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)}, \quad (3)$$

where  $TP$  is the number of true positives,  $FP$  is the number of false positives,  $TN$  is the number of true negatives, and  $FN$  is the number of false negatives. The initial parameters of the GA are shown in Table 1.

The experimental results reveal that the accuracy rates of the proposed GA-LB-LeNet are 88.8 and 98.4% for the CIA and MORPH datasets, respectively. Table 2 shows the classification accuracies of LeNet, LB-LeNet, and the proposed GA-LB-LeNet for the CIA dataset. Although LB-LeNet uses a fixed mask to reduce the number of calculations, it also reduces the classification accuracy. That is, the classification accuracy of LB-LeNet is lower than that of the original LeNet model by 3.8%. After the filter parameters are optimized through the GA, the classification accuracy of the proposed GA-LB-LeNet is 7.2% higher than that of LB-LeNet and 3.4% higher than that of LeNet. Table 3 shows the classification accuracies of LeNet, LB-LeNet, and the proposed GA-LB-LeNet for the MORPH dataset. The classification accuracy of the proposed GA-LB-LeNet is 1.1% higher than that of LB-LeNet and 0.4% higher than that of LeNet.

Table 1  
Initial parameters of GA.

|                    |      |
|--------------------|------|
| Population size    | 4600 |
| Mutation rate      | 0.05 |
| Crossover rate     | 0.8  |
| Generations        | 50   |
| Initial population | 20   |

Table 2  
Facial gender classification results using various networks for CIA dataset.

| Model       | Accuracy (%) |
|-------------|--------------|
| LeNet       | 85.40        |
| LB-LeNet    | 81.60        |
| GA-LB-LeNet | 88.80        |

Table 3  
Facial gender classification results using various networks for MORPH dataset.

| Model       | Accuracy (%) |
|-------------|--------------|
| LeNet       | 98.00        |
| LB-LeNet    | 97.30        |
| GA-LB-LeNet | 98.40        |

Tables 4 and 5 list the numbers of parameters and operations (FLOPs) required by LeNet and LB-LeNet, respectively. The convolutional layers (Conv1 and Conv2) of LeNet were replaced with the LBC layers (Conv1-1/Conv1-2 and Conv2-1/Conv2-2). Table 4 shows that the number of parameters of the convolutional layer in LeNet is 26570, compared with 5246 for LB-LeNet (Table 5), a roughly fivefold reduction. Since a large number of filter parameters in LB-LeNet are set to be fixed, we propose GA-LB-LeNet, in which the GA is used to optimize and adjust the filter parameters to obtain better parameters and improve classification results. The numbers of parameters and operations of the GA-LB-LeNet model after adding the GA are respectively shown in Tables 6 and 7. According to Table 6, the total number of parameters used in the convolutional layer (CL) for GA-LB-LeNet is around 80% less than that used in the LeNet model. In Table 7, MFLOPs represents megaFLOPs, and the ratio of the operations used in both models is also displayed. Compared with LeNet, GA-LB-LeNet requires about 30% fewer operations during convolutions. The total number of operations of the proposed GA-LB-LeNet model is 21.6% less than that of the LeNet model.

Table 4  
Parameter analysis of LeNet.

| Layer   | Parameters | Output size              | FLOPs        |
|---------|------------|--------------------------|--------------|
| Input   | 0          | $50 \times 50 \times 3$  | 0            |
| Conv1   | 1520       | $50 \times 50 \times 20$ | 7500000      |
| Pool    | 0          | $25 \times 25 \times 20$ | 0            |
| Conv2   | 25050      | $25 \times 25 \times 50$ | 6218750      |
| Pool    | 0          | $12 \times 12 \times 50$ | 0            |
| Flatten | 0          | 7200                     | 0            |
| Fc      | 3600500    | 500                      | 7199500      |
| Fc      | 1002       | 2                        | 1998         |
| Total   | 3628072    | —                        | 20.92 MFLOPs |

Table 5  
Parameter analysis of LB-LeNet.

| Layer   | Parameters | Output size              | FLOPs         |
|---------|------------|--------------------------|---------------|
| Input   | 0          | $50 \times 50 \times 3$  | 0             |
| Conv1-1 | 608        | $50 \times 50 \times 8$  | 2980000       |
| Conv1-2 | 180        | $50 \times 50 \times 20$ | 750000        |
| Pool1   | 0          | $25 \times 25 \times 20$ | 0             |
| Conv2-1 | 4008       | $25 \times 25 \times 8$  | 4995000       |
| Conv2-2 | 450        | $25 \times 25 \times 50$ | 468750        |
| Pool2   | 0          | $12 \times 12 \times 50$ | 0             |
| Flatten | 0          | 7200                     | 0             |
| Fc1     | 3600500    | 500                      | 7199500       |
| Fc2     | 1002       | 2                        | 1998          |
| Total   | 3606748    | —                        | 16.395 MFLOPs |

Table 6  
Comparison of numbers of parameters used in LeNet and GA-LB-LeNet.

| Model                           | CL parameter |      | Total CL parameters | CL parameter ratio | Total parameters |
|---------------------------------|--------------|------|---------------------|--------------------|------------------|
|                                 | BP           | GA   |                     |                    |                  |
| LeNet                           | 26570        | 0    | 26570               | —                  | 3628072          |
| GA-LB-LeNet                     | 630          | 4616 | 5246                | —                  | 3606748          |
| LeNet (baseline) vs GA-LB-LeNet | —            | —    | —                   | -80%               | —                |

Table 7  
Comparison of number of FLOPs of LeNet and GA-LB-LeNet.

| Model                           | Total CL operations | CL operation ratio | Total FLOPs   | FLOPs ratio |
|---------------------------------|---------------------|--------------------|---------------|-------------|
| LeNet                           | 13718750            | —                  | 20.920 MFLOPs | —           |
| GA-LB-LeNet                     | 9193750             | —                  | 16.395 MFLOPs | —           |
| LeNet (baseline) vs GA-LB-LeNet | —                   | -32.90%            | —             | -21.60%     |

#### 4. Conclusions

Many state-of-art networks have proved that increasing the number of layers of the CNN is a good way to improve accuracy; however, it also introduces problems such as increased storage space and computational complexity. The architectures of these networks are often limited in their application and cannot be arbitrarily used in mobile devices. In this study, the GA-LBCNN is proposed, in which the GA is used to optimize the filter mask parameters in the LBC layer in the LBCNN. With the proposed GA-LBCNN, not only can the required storage space and computational complexity be reduced, but also the best parameter combination can be found to improve the accuracy. LeNet is adopted as the basic model architecture in this study. Therefore, the proposed GA-LBCNN is also called GA-LB-LeNet. Experimental results indicate that the classification accuracy of the proposed GA-LBCNN reaches 88.8 and 98.2% for the CIA and MORPH datasets, respectively. Compared with the conventional LBCNN, the classification accuracy of the proposed GA-LBCNN is respectively increased by 7.2 and 1.1% for the two datasets. Through the data acquired from image sensors, the proposed GA-LBCNN model can not only adopt other networks as a backbone in future research but also be implemented on a field-programmable gate array to achieve real-time applications.

#### Acknowledgments

The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan, for financially supporting this research under Contract No. MOST 109-2221-E-167-027.

#### References

- 1 Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner: Proc. IEEE **86** (1998) 2278. <https://doi.org/10.1109/5.726791>
- 2 A. Krizhevsky, I. Sutskever, and G. E. Hinton: Commun. ACM **60** (2017) 84. <https://dl.acm.org/doi/10.1145/3065386>
- 3 K. Simonyan and A. Zisserman: 3rd Int. Conf. Learning Representations (ICLR 2015) 2014. <https://www.arxiv-vanity.com/papers/1409.1556/>
- 4 C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich: Proc. 2015 IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2015). <https://ieeexplore.ieee.org/document/7298594>
- 5 K. He, X. Zhang, S. Ren, and J. Sun: Proc. 2016 IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2016). <https://doi.org/10.1109/CVPR.2016.90>
- 6 H. Gao, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger: Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2017). <https://doi.org/10.1109/CVPR.2017.243>

- 7 M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi: Proc. European Conf. Computer Vision (2016) 525–546. [https://doi.org/10.1007/978-3-319-46493-0\\_32](https://doi.org/10.1007/978-3-319-46493-0_32)
- 8 Y. Zhou, L. Chen, R. Xie, L. Song, and W. Zhang: Proc. 2019 IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting (2019). <https://doi.org/10.1109/BMSB47279.2019.8971868>
- 9 B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko: Proc. 2018 IEEE Conf. Computer Vision and Pattern Recognition (2018). <https://doi.org/10.1109/CVPR.2018.00286>
- 10 S. Han, H. Mao, and W. J. Dally: Proc. Int. Conf. Learning Representations (ICLR) (2016). <https://arxiv.org/abs/1510.00149>
- 11 A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam: Proc. Conf. Computer Vision and Pattern Recognition (CVPR) (2017). <https://www.arxiv-vanity.com/papers/1704.04861/>
- 12 J. Ou and Y. Li: Neurocomputing **330** (2019) 253. <https://doi.org/10.1016/j.neucom.2018.11.028>
- 13 F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer: Proc. Int. Conf. Learning Representations (ICLR) (2016). <https://arxiv.org/abs/1602.07360>
- 14 F. J. Xu, V. N. Boddeti, and M. Savvides: Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition, (CVPR) (2017). <https://doi.org/10.1109/CVPR.2017.456>
- 15 A. Almowallad and V. Sanchez: Proc. 2020 8th Int. Workshop Biometrics and Forensics (IWBF) (2020). <https://doi.org/10.1109/IWBF49977.2020.9107940>
- 16 M. Suganuma, S. Shirakawa, and T. Nagao: Proc. 27th Int. Joint Conf. Artificial Intelligence (IJCAI) (2018). <https://doi.org/10.24963/ijcai.2018/755>
- 17 B. Baker, O. Gupta, N. Naik, and R. Raskar: Proc. Int. Conf. Learning Representations (ICLR) (2017). <https://arxiv.org/abs/1611.02167>
- 18 T. Sinha, A. Haidar, and B. Verma: Proc. 2018 IEEE Congress Evolutionary Computation (CEC) (2018). <https://doi.org/10.1109/CEC.2018.8477728>
- 19 B. Wang, Y. Sun, B. Xue, and M. Zhang: Proc. Australasian Joint Conf. Artificial Intelligence (2018) 237–250. [https://doi.org/10.1007/978-3-030-03991-2\\_24](https://doi.org/10.1007/978-3-030-03991-2_24)
- 20 B. Ma, X. Li, Y. Xia, and Y. Zhang: Neurocomputing **379** (2020) 152. <https://doi.org/10.1016/j.neucom.2019.10.007>
- 21 C. J. Lin, C. H. Lin, and S. Y. Jeng: Appl. Sci. **10** (2020) 3166. <https://doi.org/10.3390/app10093166>

## About the Authors

**Chun-Hui Lin** received her M.S. degree in computer science from the University of Texas at Dallas, Texas, USA, in 2017. Currently, she is a Ph.D. student in computer science and information engineering at National Cheng Kung University, Tainan, Taiwan. Her research interests are image processing, intelligent control, and machine/deep learning.

**Cheng-Jian Lin** received his B.S. degree in electrical engineering from Ta Tung Institute of Technology, Taipei, Taiwan, ROC, in 1986 and his M.S. and Ph.D. degrees in electrical and control engineering from National Chiao-Tung University, Taiwan, ROC, in 1991 and 1996, respectively. Currently, he is a chair professor of the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung, Taiwan, ROC, and dean of Intelligence College, National Taichung University of Science and Technology, Taichung, Taiwan, ROC. His current research interests are machine learning, pattern recognition, intelligent control, image processing, intelligent manufacturing, and evolutionary robots. (cjlin@ncut.edu.tw, cjlin@nutc.edu.tw)

**Shyh-Hau Wang** received his B.S. degree in biomedical engineering from Chung Yuan Christian University (CYCU), Chung Li, Taiwan, in 1986, his M.S. degree in biomedical engineering and electrical engineering from Drexel University, Philadelphia, PA, in 1992, and his Ph.D. degree in bioengineering from The Pennsylvania State University (PSU), University Park, PA, USA, in 1997. He worked as a research associate at the Department of Biomedical Engineering, University of Virginia, in 1997 and then as postdoctoral fellow at the NIH Ultrasound Transducer Resource Center, PSU, in 1998. Subsequently, Dr. Wang served as an assistant professor (1998), associate professor (2001), and professor (2006) at the Department of Biomedical Engineering, CYCU. Since 2009, he has been at the Department of Computer Science and Information Engineering & Institute of Medical Informatics at National Cheng Kung University, Tainan, Taiwan. His research interests are in the areas of biomedical ultrasound imaging, ultrasound tissue/material characterization, signal/image processing, medical instrumentation and informatics, and machine/deep learning.