Cerebral Microbleeds Detection via Convolutional Neural Network with and Without Batch Normalization

Jin Hong[1] and Jie Liu[1,2](✉)

[1] School of Earth Sciences and Engineering, Sun Yat-Sen University, 510275 Guangzhou, China
hongj5@mail2.sysu.edu.cn, liujie86@mail.sysu.edu.cn
[2] Guangdong Provincial Key Laboratory of Mineral Resources& Geological Processes, 510275 Guangzhou, China

Abstract. Cerebral microbleeds (CMBs) as a kind of subclinical sign are associated with cerebrovascular and cognitive diseases, as well as normal aging. Hence, it is important to detect CMBs automatically and accurately for helping medical research and preventing related diseases. CMBs can be visualized as small and rounded radiological entities via susceptibility-weighted imaging (SWI). So far, some advances about detecting CMBs automatically have been achieved. In this study, a designed CNN structure is for further improving the performance of detecting CMBs automatically. Furthermore, a breakthrough technique named batch normalization (BN), which is widely used in training deep neural networks for accelerating the training process, was tested. The performance of CNN with BN was compared to that without BN. It is found that the latter model achieved better prediction results. Afterward, four state-of-the-art methods were compared to the designed CNN. The comparison shows the designed CNN achieved the best performance with a sensitivity of 99.69%, a specificity of 96.5%, and an accuracy of 98.09%.

Keywords: Convolution neural network · Batch normalization · Cerebral microbleeds

1 Introduction

In normal or near-normal brain tissues, cerebral microbleed (CMB) means the chronic blood product appeared as small foci. They consist of hemosiderin deposits leaked from pathological blood vessels [1]. CMBs are usually associated with cerebrovascular and cognitive diseases, as well as normal aging. Research shows the existence and the distribution pattern of CMBs can indicate some underlying aetiologies caused by intra-cerebral hemorrhage (ICH) [2]. Therefore, it is important to detect CMBs within brain accurately and rapidly.

CMBs can be scanned via magnetic resonance (MR) imaging techniques which are sensitive to hemosiderin deposits. Among those techniques, susceptibility-weighted imaging (SWI) is tended to be employed to scan the brain for identifying CMBs because of the more powerful capacity of discernment comparing with traditional T1 or

© Springer Nature Singapore Pte Ltd. 2020
S. C. Satapathy et al. (eds.), *Frontiers in Intelligent Computing: Theory and Applications*, Advances in Intelligent Systems and Computing 1014,
https://doi.org/10.1007/978-981-13-9920-6_16

T2 sequence [3]. With using the magnitude and phase information, SWI can enhance the contrast [4]. Hence, we employed this technique for achieving brain images in this study.

Traditionally, radiologists with experience are employed to detect CMBs. This should be laborious and time-consuming. Because of the high inter-observer and intra-observer variability, detecting CMBs manually tend to be unreproducible. Furthermore, the CMBs with small size can be missed and CMB mimics would confuse the observers, which should make the final results error-prone. Therefore, developing automatic techniques for detecting CMBs is necessary.

Currently, many studies via computer vision techniques based on traditional machine learning have been proposed for developing computer-aided detection (CAD) systems [5–11]. Some advances about automated CMBs detection have been obtained in the last decade. Seghier et al. [12] proposed an automated microbleed detection algorithm for microbleed detection. Kuijf et al. [13] utilized the radial symmetry transform (RST) to discriminate the CMB candidates. Bian et al. [14] developed a method for detecting CMBs semi-automatically. Firstly, the 2D fast RST was employed to detect the putative CMBs. Afterward, the 3D region growing was used to exclude the false CMBs. Roy et al. [15] combined a multiple radial symmetry transform (MRST) and random forests (RF) classifier to detect CMBs. Fazlollahi et al. [16] developed a novel method named multi-scale Laplacian of Gaussian (MLG) for detecting the potential CMBs. Followed, a cascaded binary random forest (RF) was applied to identify the candidates into “Possible” and “Definite.” van den Heuvel et al. [17] utilized a two-step method for CMBs detection. Twelve features of CMBs were selected for identifying the possible CMB candidates firstly, and then the false CMBs would be removed via a random forest classifier and an object-based classifier. Hou [18] proposed a feedforward neural network containing single hidden layer for CMBs detection. They employed early stopping and an activation function named leaky rectified linear unit (ReLU) for achieving a better performance. Hou [19] employed a deep neural network (DNN) for CMBs detection. Besides the input layer and output layer, it contains four sparse auto-encoder layers and one softmax layer. Jiang [20] firstly employed the convolution neural network (CNN) method for CMBs detection. Lu [21] proposed to utilize a deep convolutional neural network.

Among the above methods, Jiang’s [20] method based on CNN achieved the best performance of CMBs detection currently. Furthermore, CNN as the most popular and advanced technique of image processing and computer vision has achieved great success in many fields [22–28]. Therefore, it is natural to believe that the CNN or its variants should be the most promising solution for CMBs detection. There have been many tricks developed for improving the performance of CNN in recent years, which may be used in our case for achieving a higher accuracy of CMBs detection. Among those tricks, batch normalization (BN) as the breakthrough technique in training deep neural networks has achieved great success in many cases [29]. Hence, we discussed the effect of batch normalization (BN) on CMBs detection in this paper.

2 Materials

We enrolled ten cerebral autosomal-dominant arteriopathy with subcortical infarcts and Leukoencephalopathy (CADASIL) patients for our study. The 3D SWI images were constructed via Syngo MR B17 software. All of them have the same size which is of $364 \times 448 \times 48$. Three neuron radiologists were employed to label the CMB voxels manually. The voxels labeled as “possible” and “definite” were regarded as CMBs, while the others were regarded as Non-CMBs. Furthermore, there are two rules for discarding the lesions: (a) they could be bleeding vessels and (b) they are bigger than 10 mm.

Afterward, we applied sliding neighborhood processing (SNP) technique to obtain the input images and their target value. A sliding window, which has a size of 61×61, was applied to sweep over the 103D brain images with stride of 1 for generating the input images. Meanwhile, low-grade samples were removed. After that, 4287 CMB and 6407 non-CMB samples were obtained. Under-sampling was applied to discard the non-CMB samples randomly for avoiding the imbalance of the samples. Finally, 4287 CMBs and 4287 Non-CMBs were achieved. Both the CMBs and the non-CMBs were divided into two equal parts: training samples containing 2144 CMBs and 2144 non-CMBs, and test samples containing 2143 CMBs and 2143 non-CMBs.

3 Methodology

Traditionally, computer vision methods can be divided into three parts: feature extraction, feature reduction, and classification. However, convolutional neural network (CNN) combines those three stages, which means CNN can extract feature and implement classification automatically [30]. Generally, a typical CNN contains convolution layer, pooling layer, and fully connected layer (FC), which is shown in Fig. 1. Among them, convolution layer is used to extract high-level and task-related features, while pooling layer is applied to reduce features. Fully connected layer can be roughly considered to be used to implement classification.

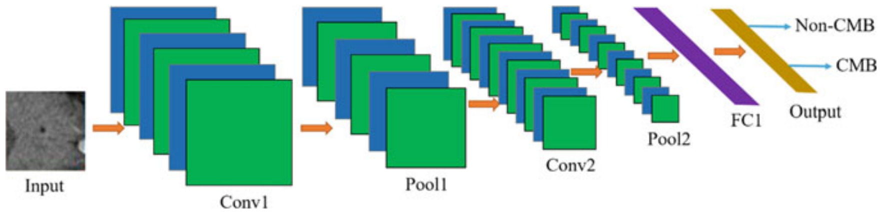

Fig. 1. A typical CNN structure

3.1 Convolution Layer

Convolution layer maybe is the most important block in CNN architecture. This layer is composed of a set of filters with learnable weights. Each filter is spatially small but has the same number of channels (also called as depth) as the input. The filters are applied to sweep over and convolve with the input for extracting features [31]. The workflow of convolution operation is shown in Fig. 2. H_{I}, W_{I}, and D denote the height, width, and depth of the input, respectively. H_{F} and W_{F} denote the height and width of the filter, respectively. H_{A} and W_{A} denote the height and width of the activation map (output), respectively. Consider N filters are employed to sweep over the input with a stride of S and padding of E, then H_{A} and W_{A} can be calculated as below.

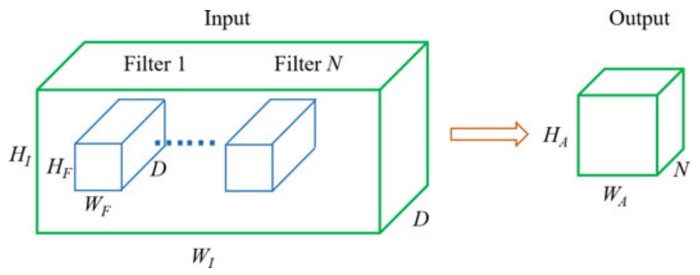

Fig. 2. Convolution layer workflow

$$H_{\mathrm{A}} = \frac{2E + H_{\mathrm{I}} - H_{\mathrm{F}}}{S} + 1 \tag{1}$$

$$W_{\mathrm{A}} = \frac{2E + W_{\mathrm{I}} - W_{\mathrm{F}}}{S} + 1 \tag{2}$$

3.2 Batch Normalization

Training deep neural networks is difficult because of the phenomenon of internal covariate shift which can change the input distribution of each layer when the neural network is being trained. Before the activation map passes through the nonlinear activation function, if the new distribution is close to the upper and lower limits of the value interval of the activation function, then the gradient of the shallow layer will be dispersed during backpropagation. This leads to the need for lower learning rates and careful parameter initialization, which slows down the training. Ioffe and Szegedy [29] proposed the idea of batch normalization (BN) for addressing the above problem. BN can convert the distribution of hidden layers' input to a standard normal distribution which falls within the sensitive interval of the activation function. This can enlarge the gradient of the shallow layer during backpropagation and accelerate the training greatly.

The procedure of normalizing the input distribution of the layer over a mini-batch [32] ([I_n], $n \in$ [1, 2, …, N]) can be illustrated as following steps:

(i) Achieve mean of the mini-batch μ

$$\mu = \frac{1}{N}\sum_{n=1}^{N} I_n \tag{3}$$

(ii) Achieve variance of the mini-batch σ^2

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N} (I_n - \mu)^2 \tag{4}$$

(iii) Achieve the normalized value of the input $\widehat{I}_n$. In order to avoid the divisor becoming zero, a small positive number φ is added to the denominator

$$\widehat{I}_n = \frac{I_n - \mu}{\sqrt{\mu^2 + \varphi}} \tag{5}$$

(iv) Achieve the outputs O_n.γ and β are two learnable parameters updated via backpropagation.

$$O_n = \gamma \widehat{I}_n + \beta \tag{6}$$

3.3 Activation Function

Generally, to activate the activation map, activation function is employed to follow the convolution layer. Furthermore, it can enhance the ability to describe the nonlinear features and accelerate training. Rectified linear unit (ReLU) was employed as the activation function. It is defined as

$$\text{ReLU}(x) = \max(0, x) \tag{7}$$

3.4 Pooling Layer

Pooling layer is used to perform features reduction for avoiding overfitting and computational burden caused by too many features. Pooling layer as a kind of nonlinear down sampling can improve CNN's nonlinear expression ability. Furthermore, it can help achieve translation invariance. In this study, max pooling [22], which is to select the max element of each pooling region, was utilized. Figure 3 shows the max pooling strategy works.

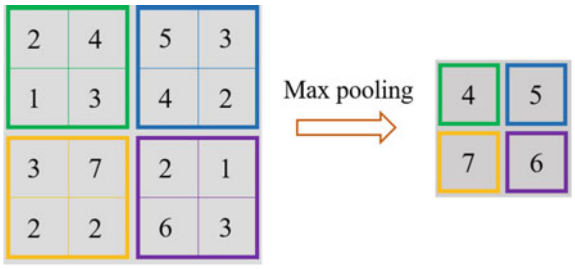

Fig. 3. A toy example of max pooling

3.5 Fully Connected Layer

After the feature extraction implemented via convolution and pooling layers, fully connected layers are used to do classification task. Each neuron is connected to all neurons which belong to the previous layer, which generates many parameters in this layer. The way of connections between neurons can be roughly illustrated as [33–36]: a neuron multiplies the input of the previous neuron by a weight matrix and then, the multiplication result is pulsed to a bias vector. Generally, the last fully connected layer is followed by a softmax function for converting the input to a probability distribution. However, the softmax function was replaced by log softmax for improving the stability of computation in this study. Suppose $[I_n]$ is the input tensor, log softmax is defined as

$$\mathrm{logsoftmax}(\boldsymbol{I})_n = \log\left(\frac{\mathrm{e}^{I_n}}{\sum_m \mathrm{e}^{I_m}}\right) \tag{8}$$

4 Results and Discussion

In this paper, we designed a six-layer CNN for CMBs detection, which contains four convolution layers and two fully connected layers. The designed CNN architecture was obtained by experience and it is shown in Table 1. In terms of training method, we employed the stochastic gradient descent with momentum of 0.9 (SGDM), which is the most common and reliable. We applied cross-entropy as the network's loss function. The initial learning rate was set to 0.01 and decreased by 5% in every 10 epochs. Mini-batch size was set to 128. The epochs were set to 100.

4.1 The Performances of CNN with and Without Batch Normalization

In view of the great success achieved by batch normalization, we tried to introduce this trick into our case. In our experiments, every convolution layer was followed by batch normalizing transform. 10 runs were carried out for enhancing the reliability of our results in view of the randomness of CNN. The training process of one run is shown in Fig. 4a. The prediction results are given in Table 2. Meanwhile, the performance of CNN without batch normalization as the reference was shown in Fig. 4b and Table 3.

Table 1. Designed CNN

Layer index	Layer type	Hyper parameters
1	Convolution	16 (Out Channels), 7 (Kernel Size), 1 (Stride), 3 (Padding)
	Pooling	5 (Kernel Size), 5 (Stride), 0 (Padding)
2	Convolution	32, 3, 1, 1
	Pooling	3, 3, 0
3	Convolution	64, 3, 1, 1
	Pooling	2, 2, 0
4	Convolution	128, 3, 1, 1
	Pooling	2, 2, 0
5	Fully connected	64 × 128 (Weights), 64 × 1 (Bias)
6	Fully connected	2 × 64, 2 × 1

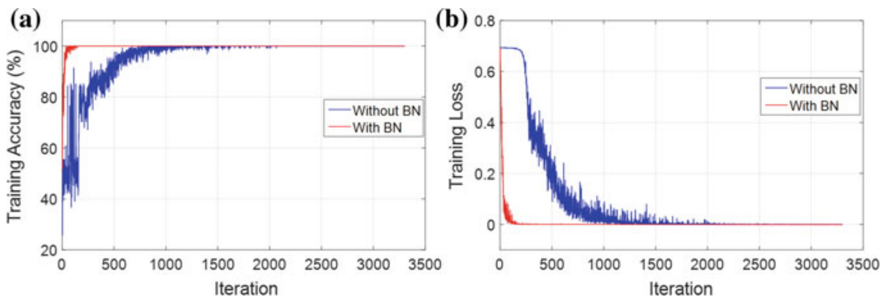

Fig. 4. Training performance of one run

Table 2. Prediction results of CNN with batch normalization

Run	Sensitivity (%)	Specificity (%)	Accuracy (%)
1st run	90.39	97.62	94.00
2nd run	97.67	97.06	97.36
3rd run	89.64	98.13	93.89
4th run	93.84	98.13	95.99
5th run	91.88	97.85	94.87
6th run	92.63	97.76	95.19
7th run	95.19	97.53	96.36
8th run	91.93	97.57	94.75
9th run	96.13	97.95	97.04
10th run	97.06	97.48	97.27
Std	2.65	0.31	1.25
Average	93.64	97.71	95.67

Table 3. Prediction results of CNN without batch normalization

Run	Sensitivity (%)	Specificity (%)	Accuracy (%)
1st run	99.81	96.69	98.25
2nd run	99.02	96.13	97.57
3rd run	99.86	96.59	98.23
4th run	99.72	96.41	98.06
5th run	99.63	96.17	97.90
6th run	100.00	96.69	98.34
7th run	99.63	96.36	97.99
8th run	99.67	97.06	98.37
9th run	99.63	96.17	97.90
10th run	99.91	96.69	98.30
Std	0.25	0.28	0.24
Average	99.69	96.50	98.09

Figure 4 shows that the training loss and accuracy of CNN with batch normalization converges faster than that without batch normalization. This means that the batch normalization did accelerate the training. However, a faster training process does not mean a higher accuracy of prediction. As the Tables 2 and 3 show, CNN with batch normalization achieved abetter average specificity of 97.71%, while CNN without batch normalization achieved a better average sensitivity of 99.69% and a better average accuracy of 98.09%. This denotes that the CNN without batch normalization performed better than that with batch normalization for CMBs detection in view of the fact that sensitivity and accuracy are more important than specificity in clinical routine. Furthermore, the standard deviation of prediction results of CNN without batch normalization is smaller than that with batch normalization, which means the former model has better stability. In summary, we suggested abandoning batch normalization in our case.

4.2 Compared to State-of-the-Art Methods

Furthermore, we compared the designed CNN without batch normalization to four state-of-the-art methods, which is shown in Table 4. It is found that the designed CNN achieved the best performance with a sensitivity of 99.69%, a specificity of 96.5%, and an accuracy of 98.09%, since sensitivity and accuracy are more important than specificity.

Table 4. Comparison with state-of-the-art approaches

Method	Sensitivity (%)	Specificity (%)	Accuracy (%)
MRST + RF [15]	85.7	99.5	–
SNP + SLFN + LReLU [18]	93.05	93.06	93.06
7-layer SAE [19]	95.13 ± 0.84	93.33 ± 0.84	94.23 ± 0.84
7-layer CNN [20]	96.94	97.18	97.18
The designed CNN	99.69	96.50	98.09

5 Conclusion

In this study, a six-layer CNN was designed for detecting CMBs. The performance of the CNN with batch normalization was compared to that without batch normalization. It is found that the latter model achieved better prediction results, which means batch normalization did not help improving the performance of CNN in our case. Afterward, we compared the designed CNN to four state-of-the-art methods. The comparison shows our designed CNN achieved the best performance with a sensitivity of 99.69%, a specificity of 96.5%, and an accuracy of 98.09%.

That batch normalization failed in our case surprised us in view of the success in many other tasks achieved by batch normalization. We cannot give an accurate explanation for this now but more works will be done for exploring the reasons in the future. Compared with traditional machine learning methods, CNN is like a "black box" which cannot show the reasons for how and why it works. However, in view of the fact that CNN performs better in CMBs detection than traditional machine learning, we will still focus on CNN method or its variants in future [37]. We will optimize the CNN structure or try more combinations of hyper parameters for further improving the accuracy. Furthermore, more data will be collected for testing our approach.

Acknowledgements. We have obtained the permission to use the data/image/sources from the competent authorities. We take responsibility for the same. Authors are thankful to Guangdong Provincial Key Laboratory of Mineral Resources and Geological Processes, Guangzhou, China for their support.

References

1. Greenberg, S.M., Vernooij, M.W., Cordonnier, C., Viswanathan, A., Salman, R.A.S., Warach, S., Launer, L.J., Van Buchem, M.A., Breteler, M.M.B., Microbleed Study G: Cerebral microbleeds: a guide to detection and interpretation. Lancet Neurol. **8**(2), 165–174 (2009)
2. Charidimou, A., Shakeshaft, C., Werring, D.J.: Cerebral microbleeds on magnetic resonance imaging and anticoagulant-associated intracerebral hemorrhage risk. Front. Neurol. **3**, 133 (2012)
3. Nandigam, R., Viswanathan, A., Delgado, P., Skehan, M., Smith, E., Rosand, J., Greenberg, S., Dickerson, B.: MR imaging detection of cerebral microbleeds: effect of susceptibility-weighted imaging, section thickness, and field strength. Am. J. Neuroradiol. **30**(2), 338–343 (2009)
4. Mittal, S., Wu, Z., Neelavalli, J., Haacke, E.M.: Susceptibility-weighted imaging: technical aspects and clinical applications, part 2. Am. J. Neuroradiol. **30**(2), 232–252 (2009)
5. Wu, L.: Weights optimization of neural network via improved BCO approach. Progr. Electromagnet. Res. **83**, 185–198 (2008). https://doi.org/10.2528/PIER08051403
6. Wu, L.N.: Pattern recognition via PCNN and tsallis entropy. Sensors **8**(11), 7518–7529 (2008). https://doi.org/10.3390/s8117518
7. Zhang, Y.: Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network. Expert Syst. Appl. **36**(5), 8849–8854 (2009)

8. Naggaz, N.: Remote-sensing image classification based on an improved probabilistic neural network. Sensors **9**(9), 7516–7539 (2009)
9. Wei, G.: Color image enhancement based on HVS and PCNN. Sci. China Inf. Sci. **53**(10), 1963–1976 (2010). https://doi.org/10.1007/s11432-010-4075-9
10. Wu, L.: Classification of fruits using computer vision and a multiclass support vector machine. Sensors **12**(9), 12489–12505 (2012). https://doi.org/10.3390/s120912489
11. Ji, G.: Fruit classification using computer vision and feedforward neural network. J. Food Eng. **143**, 167–177 (2014). https://doi.org/10.1016/j.jfoodeng.2014.07.001
12. Seghier, M.L., Kolanko, M.A., Leff, A.P., Jager, H.R., Gregoire, S.M., Werring, D.J.: Microbleed detection using automated segmentation (MIDAS): a new method applicable to standard clinical MR images. Plos One **6**(3), Article ID: e17547 (2011). https://doi.org/10.1371/journal.pone.0017547
13. Kuijf, H.J., de Bresser, J., Geerlings, M.I., Conijn, M.M.A., Viergever, M.A., Biessels, G.J., Vincken, K.L.: Efficient detection of cerebral microbleeds on 7.0 T MR images using the radial symmetry transform. Neuroimage **59**(3), 2266–2273 (2012). https://doi.org/10.1016/j.neuroimage.2011.09.061
14. Bian, W., Hess, C.P., Chang, S.M., Nelson, S.J., Lupo, J.M.: Computer-aided detection of radiation-induced cerebral microbleeds on susceptibility-weighted MR images. Neuroimage-Clinical **2**, 282–290 (2013). https://doi.org/10.1016/j.nicl.2013.01.012
15. Roy, S., Jog, A., Magrath, E., Butman, J.A., Pham, D.L.: Cerebral microbleed segmentation from susceptibility weighted images. In: Proceedings of SPIE, 9413, Article ID: 94131E (2015). https://doi.org/10.1117/12.2082237
16. Fazlollahi, A., Meriaudeau, F., Giancardo, L., Villemagne, V.L., Rowe, C.C., Yates, P., Salvado, O., Bourgeat, P.: Computer-aided detection of cerebral microbleeds in susceptibility-weighted imaging. Comput. Med. Imaging Graph. **46**(Part 3), 269–276 (2015). https://doi.org/10.1016/j.compmedimag.2015.10.001
17. van den Heuvel, T.L.A., van der Eerden, A.W., Manniesing, R., Ghafoorian, M., Tan, T., Andriessen, T.M.J.C., Vande Vyvere, T., van den Hauwe, L., ter Haar Romeny, B.M., Goraj, B.M., Platel, B.: Automated detection of cerebral microbleeds in patients with traumatic brain injury. NeuroImage: Clinical **12**, 241–251 (2016). http://doi.org/10.1016/j.nicl.2016.07.002
18. Hou, X.-X.: Voxelwise detection of cerebral microbleed in CADASIL patients by leaky rectified linear unit and early stopping. Multimedia Tools Appl. **77**(17), 21825–21845 (2018). https://doi.org/10.1007/s11042-017-4383-9
19. Hou, X.-X.: Seven-layer deep neural network based on sparse autoencoder for voxelwise detection of cerebral microbleed. Multimedia Tools Appl. **77**(9), 10521–10538 (2018). https://doi.org/10.1007/s11042-017-4554-8
20. Jiang, Y.Y.: Cerebral micro-bleed detection based on the convolution neural network with rank based average pooling. IEEE Access **5**, 16576–16583 (2017). https://doi.org/10.1109/access.2017.2736558
21. Lu, S.: Detection of cerebral microbleeding based on deep convolutional neural network. In: 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China. IEEE, pp. 93–96 (2017). https://doi.org/10.1109/iccwamtip.2017.8301456
22. Sui, Y.X.: Classification of Alzheimer's disease based on eight-layer convolutional neural network with leaky rectified linear unit and max pooling. J. Med. Syst. **42**(5), Article ID: 85 (2018). https://doi.org/10.1007/s10916-018-0932-7
23. Pan, C.: Multiple sclerosis identification by convolutional neural network with dropout and parametric ReLU. J. Comput. Sci. **28**, 1–10 (2018). https://doi.org/10.1016/j.jocs.2018.07.003

24. Pan, C.: Abnormal breast identification by nine-layer convolutional neural network with parametric rectified linear unit and rank-based stochastic pooling. J. Comput. Sci. **27**, 57–68 (2018). https://doi.org/10.1016/j.jocs.2018.05.005
25. Tang, C.: Twelve-layer deep convolutional neural network with stochastic pooling for tea category classification on GPU platform. Multimedia Tools Appl. **77**(17), 22821–22839 (2018). https://doi.org/10.1007/s11042-018-5765-3
26. Lv, Y.-D.: Alcoholism detection by data augmentation and convolutional neural network with stochastic pooling. J. Med. Syst. **42**(1), Article ID: 2 (2018)
27. Muhammad, K.: Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation. Multimedia Tools Appl. **78**(3), 3613–3632 (2017). https://doi.org/10.1007/s11042-017-5243-3
28. Zhao, G.: Polarimetric synthetic aperture radar image segmentation by convolutional neural network using graphical processing units. J. Real-Time Image Process. (2017). https://doi.org/10.1007/s11554-017-0717-0
29. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift (2015). arXiv preprint arXiv:150203167
30. Jia, W.-J.: Ford motorcar identification from single-camera side-view image based on convolutional neural network. In: 18th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), Guilin, China, pp. 173–180. Springer (2017)
31. Das, R., Piciucco, E., Maiorana, E., Campisi, P.: Convolutional neural network for finger-vein-based biometric identification. IEEE Trans. Inf. Forensics Secur. **14**(2), 360–373 (2019). https://doi.org/10.1109/tifs.2018.2850320
32. Barik, A., Rai, R.K., Chowdhury, A.: Alcohol use-related problems among a rural indian population of West Bengal: an application of the alcohol use disorders identification test (AUDIT). Alcohol Alcoh. **82**(2), 215–223 (2016). Oxford Academic
33. Li, Y.-J.: Single slice based detection for Alzheimer's disease via wavelet entropy and multilayer perceptron trained by biogeography-based optimization. Multimedia Tools Appl. **77**(9), 10393–10417 (2018). https://doi.org/10.1007/s11042-016-4222-4
34. Sun, Y.: A multilayer perceptron based smart pathological brain detection system by fractional fourier entropy. J. Med. Syst. **40**(7), Article ID: 173 (2016). https://doi.org/10.1007/s10916-016-0525-2
35. Wei, L.: Fruit classification by wavelet-entropy and feedforward neural network trained by fitness-scaled chaotic ABC and biogeography-based optimization. Entropy **17**(8), 5711–5728 (2015). https://doi.org/10.3390/e17085711
36. Wu, J.: Fruit classification by biogeography-based optimization and feedforward neural network. Expert Syst. **33**(3), 239–253 (2016). https://doi.org/10.1111/exsy.12146
37. Dodge, S.F., Karam, L.J.: Quality robust mixtures of deep neural networks. IEEE Trans. Image Process. **27**(11), 5553–5562 (2018). https://doi.org/10.1109/tip.2018.2855966