IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 29, 2018, accepted December 5, 2018, date of publication December 10, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2885639

A Modified Deep Convolutional Neural Network for Abnormal Brain Image Classification

D. JUDE HEMANTH[1], J. ANITHA[1], ANTOANELA NAAJI[2], OANA GEMAN[3], DANIELA ELENA POPESCU[4], AND LE HOANG SON[5,6]

[1]Department of ECE, Karunya Institute of Technology and Sciences, Coimbatore 641114, India
[2]Faculty of Economics, Engineering and Computer Science, Vasile Goldiș Western University of Arad, 389360 Arad, Romania
[3]Department of Health and Human Development, Ștefan cel Mare University of Suceava, 720229 Suceava, Romania
[4]Department of Computers and Information Technology, University of Oradea, 410087 Oradea, Romania
[5]Division of Data Science, Ton Duc Thang University, Ho Chi Minh City 700000, Vietnam
[6]Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City700000, Vietnam

Corresponding author: Le Hoang Son (lehoangson@tdtu.edu.vn)

ABSTRACT Deep learning techniques have gained significant importance among artificial intelligence techniques for any computing applications. Among them, deep convolutional neural networks (DCNNs) is one of the widely used deep learning networks for any practical applications. The accuracy is generally high and the manual feature extraction process is not necessary in these networks. However, the high accuracy is achieved at the cost of huge computational complexity. The complexity in DCNN is mainly due to: 1) increased number of layers between input and output layers and 2) two set of parameters (one set of filter coefficients and another set of weights) in the fully connected network need to be adjusted. In this paper, the second aspect is targeted to reduce the computational complexity of conventional DCNN. Suitable modifications are performed in the training algorithm to reduce the number of parameter adjustments. The weight adjustment process in the fully connected layer is completely eliminated in the proposed modified approach. Instead, a simple assignment process is used to find the weights of this fully connected layer. Thus, the computational complexity is significantly reduced in the proposed approach. The application of modified DCNN is explored in the context of magnetic resonance brain tumor image classification. Abnormal brain tumor images from four different classes are used in this paper. The experimental results show promising results for the proposed approach.

INDEX TERMS Deep learning, convolutional neural network, brain images, image classification.

I. INTRODUCTION

Deep learning approaches are one of the prime computational intelligence techniques used for medical imaging applications. Specifically, deep learning approaches are widely used for medical image classification which falls under pattern recognition applications. These deep learning based medical image classification approaches are normally employed in automated disease diagnostic systems. The main method among the deep learning approaches is the Deep Convolutional Neural Network (DCNN). The main advantage of DCNN is the high accuracy which is achieved with the help of many layers and automated feature extraction process. However, the high accuracy is achieved at the cost of high computational complexity. It is a well-defined concept that a system must be efficient in terms of accuracy and complexity for real time applications. Literature survey reveals several DCNN based research works for medical image classification applications.

Deep neural network-based brain tumor image classification is proposed in [1]. Three types of abnormal brain image category are used in this work. The conventional training process is used to classify images. The application of DCNN for Computer Tomography (CT) brain image classification is explored in [2]. The fusion of 2D CNN and 3D CNN is exploited in this work for performance enhancement of the conventional method. Early detection of Alzheimer is the focus of this work. DCNN is also used for Alzheimer disease detection in [3]. Positron Emission Tomography (PET) brain images are classified using DCNN for the disease detection process. Bi-level classification is carried out in this work. A software for deep learning based medical image processing is developed by Eli *et al.* [4]. Medical image classification is

the prime focus of this work and it is open-source software. A detailed survey on medical image analysis using deep learning approaches is given in [5]. The pros and cons of DCNN are discussed in detail in this work.

An improved deep learning approach based on human visual perception is proposed for image classification in [6]. The drawback of the conventional method and suggestions for improvement is available in this work. Deep autoencoder neural network-based functional MRI (f-MRI) brain image classification is proposed in [7]. This method is used for the accurate prediction of schizophrenia. Another survey on deep learning algorithms for biomedicine applications is available in [8]. DCNN and Deep Neural Networks are dealt in detail for various medical imaging applications. Several other deep learning architectures are also discussed in this work. A modified Deep Neural network for pattern recognition is implemented in [9]. The modifications are performed in such a way to reduce the number of training images. This method can be extended for any practical applications. Deep Residual Network based medical image classification is proposed in [10]. Four different types of abnormal categories are sued in this work. However, the complexity is quite high due to the large number of layers used in this work.

Autism disorder classification using Deep neural network is explored in [11]. Only bi-level (normal/abnormal) classification is carried out in this work. However, different stages of autism classification are necessary for practical applications. A modified DCNN is proposed in [12]. Modifications are done in such a way that large dataset is not necessary for training the proposed approach. Classification accuracy is used as the performance measure for analysing this method. Classification of multimodal medical images using deep convolutional neural network is proposed in [13]. This method also emphasizes on achieving high accuracy for the proposed approach. Glioma tumor classification and segmentation using deep convolutional neural network is illustrated in [14]. Five different DCNN based approaches are proposed in this work. Sensitivity and Specificity are the performance measures used in this work. Deep convolutional neural networks are also used for detection and diagnosis of seizures [15]. Deep convolutional neural networks for brain tumor detection is also explored in [16]. Grading of meningioma from MR images using deep convolutional neural network is developed in [17]. DCNN is also used for classification of other medical images [18]. Few modified deep neural networks for image classification are proposed in [19] and [20].

In this work, a modified DCNN is proposed for abnormal brain image classification. The modification is performed in the fully connected layer of conventional DCNN. The weights in the fully connected layer are estimated by an assignment process rather than the gradient descent mode of training used in conventional DCNN. This methodology reduces the computational complexity to a higher extent without compromising the accuracy. The rest of the proposed MDCNN is same as that of the conventional DCNN. Experiments are conducted on real-time MR brain tumor images. The performance of the proposed approach is analysed in terms of accuracy and complexity. The rest of the paper is organized as: Section 2 covers the materials and methods, Section 3 deals with the conventional CNN, Section 4 deals with the proposed modified DCNN, Section 5 covers the experimental results and Section 6 provides the conclusions and future scope of this paper.

II. MATERIALS AND METHODS

The proposed methodology used in the work is shown in Figure 1.

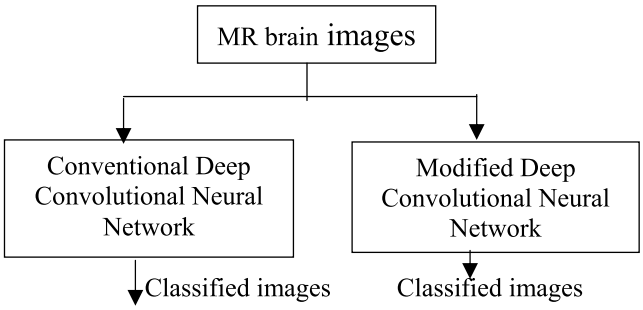

FIGURE 1. Proposed Methodology.

Abnormal MR brain tumor images collected from M/s. Devaki Scan Centre are used in this work. These images are taken from four abnormal tumor categories such as Metastasis, Meningioma, Glioma and Astrocytoma. All the images are grey scale images with size of $256 * 256$. The total number of images used in this work is 220 which includes T1, T2 and T2 flair images. Since DCNN recognizes the images based on shapes, images are chosen with unique tumor shapes for each abnormal category. Sample images are shown in Figure 2.

These images are directly given to the conventional DCNN and modified DCNN. The output of DCNN and MDCNN are the classified images. These approaches are dealt in detail in the next section.

III. CONVENTIONAL DEEP CONVOLUTIONAL NEURAL NETWORK

The architecture and the training algorithm of DCNN are discussed in this section.

A. ARCHITECTURE OF DCNN

The architecture of the DCNN used in this work is shown in Figure 3.

In the above figure 'aa' corresponds to the MR brain input images and 'bb' corresponds to the classified images. The three major modules are convolutional layer, max-pool layer and fully connected layer. The details of these modules are discussed in subsequent sections.

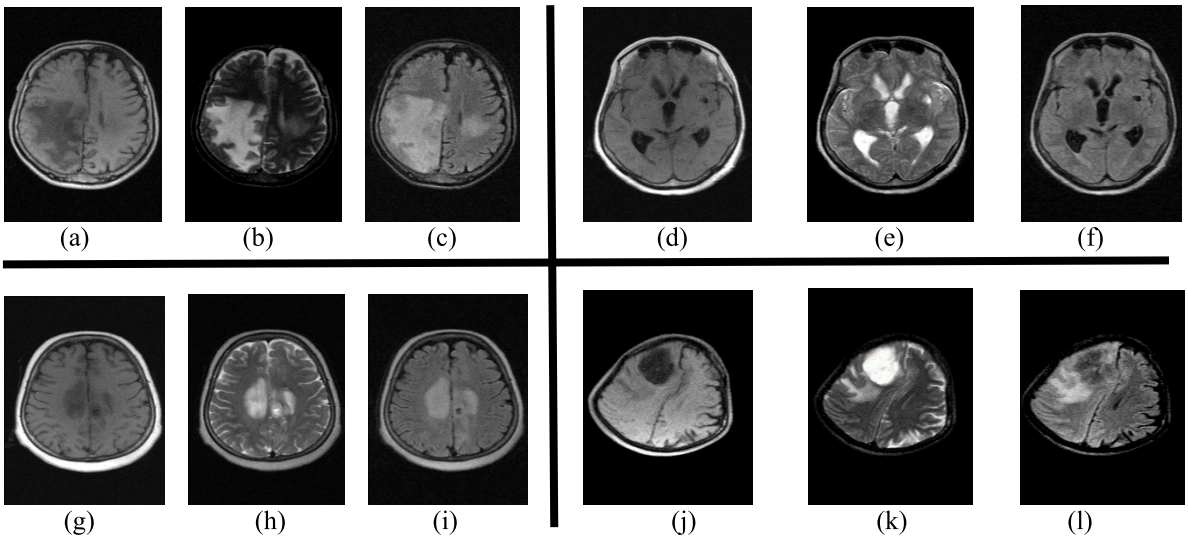

FIGURE 2. Sample input images: (a) T1-Metastasis, (b) T2-Metastasis, (c) T2flair – Metastasis, (d) T1-Meningioma, (e) T2-Meningioma, (f) T2flair-Meningioma, (g) T1-Glioma, (h) T2-Glioma, (i) T2flair-Glioma, (j) T1-Astrocytoma, (k) T2-Astrocytoma and (l) T2flair-Astrocytoma.

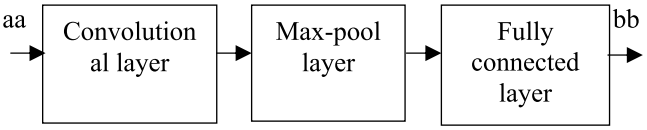

FIGURE 3. Architecture of DCNN.

1) CONVOLUTIONAL LAYER

Three convolution layers are used in succession in the proposed architecture. The first convolution layer is used to extract the low-level features from the input images. The output of the second and the third convolution layer provides the higher-level features. There are two inputs for each convolution layer. The first input is the intensity values of the input images and the second input is the filter coefficients. The filter coefficient is also called as weights. These are the trainable parameters in the input layer. The internal architecture of convolutional layer is shown in Figure 4.

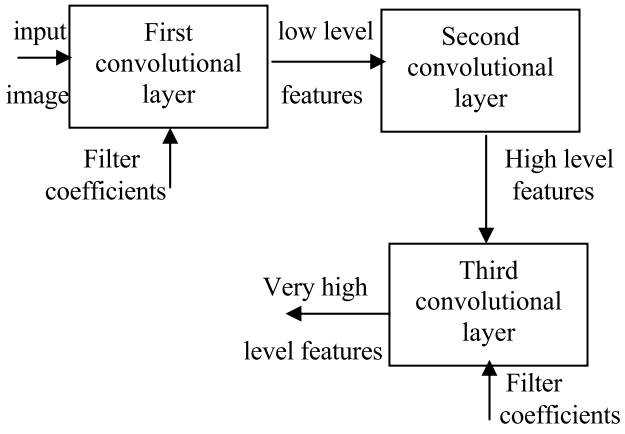

FIGURE 4. Arrangement of convolutional layers.

The complete process within these layers are given in the following algorithm.

Step 1: Initialize the size of the filter and randomly set the filter coefficients. The size of the filter used in this work is $[4 \times 4 \times 1]$. Here, the third parameter '1' shows that the input image is a grey level image.

There is no overlapping between the pixels during the sliding of filter coefficients. Zero padding is also not necessary for the convolution process used in this work.

Step 2: Perform convolution between the input image and the filter coefficients using the following formula. The convolution operation can be implemented as the dot product between the input and the filter coefficients (or) the weights.

$$y(m, n) = \sum_{i,j=1tok} x(i, j) * w(i, j) \quad (1)$$

where y = output matrix
x = input matrix
w = filter coefficients (or) weights

In this work, k = 4, m = size of the input image in row-wise arrangement/size of the weight matrix in row-wise arrangement, n = size of the input image in column-wise arrangement/size of the weight matrix in column-wise arrangement. Thus, the output matrix yields a 2-D data with dimension of $[64 \times 64]$.

Step 3: In this work, 12 filters are used and hence the same process is repeated for all the filters. The overall output is $[64 \times 64 \times 12]$.

Step 4: Each filter output is stacked in the column-wise direction to generate the overall volume of the output data. Each filter output represents low level features such as a line, curve, etc.

Step 5: The second convolutional layer works on the output of the first convolutional layer with a new set of

filter coefficients. The same process is repeated and hence the output dimension is [16 × 16 × 12]. However, these outputs represent a better shape such as an extended curve, semi-circle, etc.

Step 6: The same process is repeated with the third convolutional layer. The size of the output layer is [4 × 4 × 12]. A better shape of the input image is visible with the output of the third layer.

However, the selection of the filter co-efficient plays a major role in the success rate of the convolution process. These parameters are trainable which will be discussed later in the manuscript. The total number of filter co-efficient for all the three convolutional layers are [4 × 4 × 12 × 3]. It may be noted that the trainable parameters are large enough to increase the complexity of the system.

2) RELU LAYER

The ''ReLu layer'' stands for ''Rectified Linear Unit layer''. This layer employs simple rectified linear function after every convolutional layer. The main function of this layer is to eliminate any zero and negative values. This will enhance the contrast of the output of each convolutional layer which yields better representation of the features. The formula used is given by:

$$f(x) = \max(0, x) \tag{2}$$

3) MAX-POOL LAYER

A single max-pool layer is used after the third convolutional layer. It is used to downsize the data. The entire [4 × 4] output data is grouped into 4 clusters with [2 × 2] size for each filter. The maximum value among a [2 × 2] group will be chosen. Thus, the output is down sampled by 4 which yields an output value of [2 × 2] for a single filter. The same process is repeated for all the filters which results in an output value of [2 × 2 × 12].

4) FULLY CONNECTED LAYER

This layer is the final layer of the convolutional neural network. This is the decision-making layer of the DCNN. More emphasis is given for this layer since the modifications are performed in this layer in the modified approach. In this work, a 3-layer architecture is used with one input layer, one hidden layer and one output layer. The number of neurons in the input layer is 48 corresponding to the output of max-pool layer. The number of neurons in the output layer is 4 which corresponds to the number of output classes. 60 neurons are used in the hidden layer. Two set of weight matrices are used in this network. One set of weight matrix is seen between the input and hidden layer with dimension [48 × 60]. Another set of weight matrix is available between the hidden layer and the output layer [60 × 10]. The architecture of fully connected layer is shown in Figure 5.

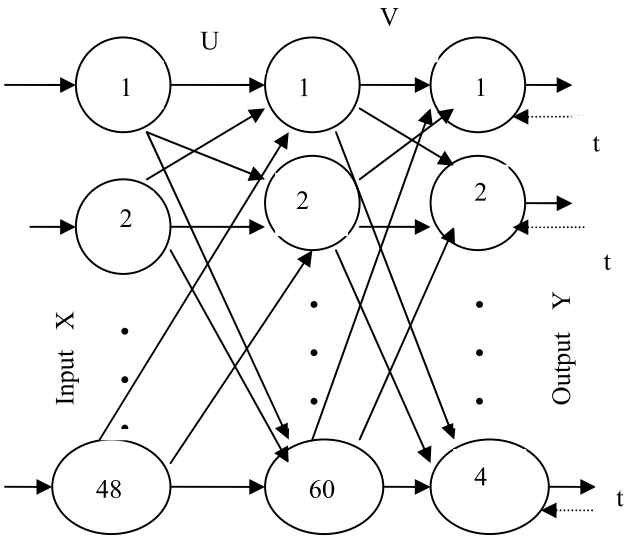

FIGURE 5. Architecture of fully connected layer.

B. TRAINING ALGORITHM OF DCNN

In DCNN, two set of parameters need adjustment during the training process. One set is the weights associated with the fully connected layer and another set of parameters is the filter coefficients (weights) available in the convolution layers. Stochastic gradient descent algorithm is used in this work for the training process. The training algorithm is carried out in two passes: (a) Forward Pass and Reverse Pass. The entire training algorithm is summarized below:

1) FORWARD PASS

Step 1: The convolution between the randomly initialized filter coefficients and the inputs are estimated using Equation (1). This process is repeated for all the three convolutional layers. It may be noted that ReLu activation function is applied after each convolutional layer.

Step 2: The down sampling of the output of convolutional layer is performed using the max pool layer.

Step 3: The 2-D output data is now converted into a single column data and applied to the input layer of the fully connected neural network.

Step 4: Randomly initialize the weight matrices for the hidden layer and the output layer. The weight matrix between the input and hidden layer is 'U' and the weight matrix between the hidden layer and output layer is 'V'. Estimate the *NET* value of the hidden layer neurons using the following formula:

$$z_{net} = \sum XU \tag{3}$$

where z_{net} corresponds to the *NET* value of the hidden layer and X corresponds to the inputs.

Step 5: Estimate the output value of hidden layer by passing through sigmoidal activation function.

$$z = \frac{1}{1 + e^{-z_{net}}} \tag{4}$$

where z corresponds to the output of the hidden layer neurons.

Step 6: Calculate the *NET* value of the output layer using the following formula:

$$y_{net} = \sum zV \tag{5}$$

where y_{net} corresponds to the NET value of the output layer.

Step 7: Estimate the output value of output layer using the following formula

$$y = \frac{1}{1 + e^{-y_{net}}} \tag{6}$$

where y corresponds to the output of the fully connected neural network.

Step 8: Cross entropy function is used as the cost function in this work. The cost value is estimated using the following formula:

$$E = -\frac{1}{N}\sum_{n=1}^{N}[t_n \log(y_n) - (1 - t_n)\log(1 - y_n)] \tag{7}$$

where t corresponds to the target of the fully connected network.

2) REVERSE PASS

The weights of the fully connected neural network and the weights (filter co-efficient) are adjusted in the reverse pass based on the cost value. Since the error is propagated in the backwards direction, the weights of the fully connected layer are adjusted first followed by the weights of the convolutional layers.

Step 9: Adjust the weights between the output layer and the hidden layer using the following formulae:

$$V_{jk}\,(new) = V_{jk}\,(old) + \left(\Delta V_{jk}\right) \tag{8}$$

$$\Delta V_{jk} = \propto \delta_k z_j \tag{9}$$

$$\delta_k = (t_k - y_k) f^{|}(y_{net}) \tag{10}$$

where j corresponds to the index of hidden layer, k corresponds to the index of output layer and $\propto$ corresponds to the learning rate. The value of learning rate used in this work is 0.6.

Step 10: Adjust the weights between the hidden layer and input layer using the following formulae:

$$U_{ij}\,(new) = U_{ij}\,(old) + (\Delta U_{ij}) \tag{11}$$

$$\Delta U_{ij} = \propto \delta_j x_i \tag{12}$$

$$\delta_j = \delta_{inj} f^{|}(z_{net}) \tag{13}$$

$$\delta_{inj} = \sum \delta_k V_{jk} \tag{14}$$

It is clearly evident that the number of steps required for training increases as we move away from the output layer.

Step 11: Adjust the weights of the third, second and first convolutional layer using the same process as mentioned in step 9 and step 10. Chain rule differentiation method shall be adopted during the training process. However, it may be noted that the computational complexity increases with increase in the number of layers.

Step 12: After adjusting the weights of the convolutional layers, the forward pass steps are once again carried out in an iterative manner. The entire process is repeated till the cost value reaches below a specified value.

Step 13: The testing process is then carried out with unknown images and the images are categorized based on the output values of neurons in the fully connected layer.

IV. MODIFIED DEEP CONVOLUTIONAL NEURAL NETWORK

In this work, modifications are performed in the training algorithm of DCNN to make it more efficient in terms of accuracy and complexity. Conventionally, there are two parameters (one in the convolutional layer and other in the fully connected layer) in DCNN which need to be adjusted. However, in the proposed approach, the parameter in the convolutional layer alone shall be adjusted. Suitable modifications are made in the fully connected layer to estimate the weight values without any iteration. The architecture of the MDCNN is same as that of the DCNN. The training algorithm of the proposed MDCNN is given below.

A. TRAINING ALGORITHM OF MDCNN

Step 1: Implement steps 1-3 as discussed in training algorithm of DCNN.

Step 2: Fix the cost value in Equation (7). The ideal value of cross entropy is zero. However, it is not practically possible. Even in conventional DCNN, the algorithm is assumed to be converged when it reaches a higher value than zero. Hence, in this approach, this value is fixed at 0.01 which is the most commonly used value in the literature.

Step 3: Since cross entropy value and the target value are known, the output value y is estimated using Equation (7).

Step 4: With the estimated output value of output layer, the NET value of the output layer can be estimated using Equation (6).

Step 5: Now, the objective is to find the output layer weights using Equation (5). However, the output value of the hidden layer (z) is unknown. It can be determined using Equation (9) with the simple assumption on the difference between the weights (ΔV_{jk}). During the converged state, the difference between any two weight values must be very minimum. It cannot be zero which is an ideal condition. Hence, it is fixed as 0.01. Now, the output value of hidden layer (z) can be estimated. Using this value and the estimated NET value of output layer, the weights of the output layer is estimated using Equation (5).

Step 6: The NET value of the hidden layer is estimated using Equation (4).

Step 7: Since the input and the NET value are known, the weights of the hidden layer are estimated using Equation (3).

Step 8: The rest of the training process remains the same as conventional DCNN.

Thus, the weight values are estimated without any iterations. It has been estimated with simple mathematical

process. It may be noted that this process is done for only one iteration which reduces the computational complexity to high extent. Now, only the filter coefficients in the convolutional layers need adjustment. In the MDCNN, the complex weight adjustment equations are not necessary which improves the practical feasibility of the proposed approach.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

The dataset used in this work is discussed under section 2. However, the dataset is initially divided into training dataset and testing dataset. The performance measures are estimated only for the testing images. Table 1 shows the details of training and testing dataset used in this work.

TABLE 1. Brain image database.

Category	Training images	Testing images
Metastasis	20	52
Meningioma	20	30
Glioma	20	30
Astrocytoma	20	28

The performance measures used in this work are classification accuracy, True Positive Rate (TPR), True Negative Rate (TNR) and computational complexity. The accuracy measures are estimated using the following formulae:

$$ClassificationAccuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (15)$$

$$TruePositiveRate = \frac{TP}{TP + FN} \quad (16)$$

$$TrueNegativeRate = \frac{TN}{TN + FP} \quad (17)$$

where TP = True Positive; TN = True Negative; FP = False Positive and FN = False Negative.

A. ACCURACY MEASURES OF THE CLASSIFIERS

The accuracy measures include Classification accuracy, True positive rate and True negative rate. Initially, the TP, TN, FP and FN values are estimated from the output values of neurons in the fully connected layer. These values are then used to estimate the performance measures using the formulae mentioned above. Table 2 and Table 3 shows the confusion matrix of the proposed classifiers. The confusion matrix provides the details about the TP, TN, FP and FN values.

TABLE 2. Confusion matrix of the conventional DCNN.

Category	Me	Men	Glioma	Astrocytoma
Metastasis	48	2	1	1
Meningioma	1	27	1	1
Glioma	2	0	26	2
Astrocytoma	1	1	2	24

Me=Metastasis; Men=Meningioma

If a specific neuron in the fully connected layer yields a higher value for a specific input testing image, then that input is assigned to the pre-defined category of that neuron. The same process is repeated for the values given in the above-mentioned tables. Table 4 and Table 5 shows the performance measures of these proposed approaches.

TABLE 3. Confusion matrix of the conventional MDCNN.

Category	Me	Men	Glioma	Astrocytoma
Metastasis	49	1	1	1
Meningioma	1	28	0	1
Glioma	0	0	28	2
Astrocytoma	1	1	1	25

Me=Metastasis; Men=Meningioma

It is evident that the accuracy measures are better for the proposed approach in comparison with the conventional DCNN. One of the significant reasons is that the training process using the conventional "backprop" algorithm is eliminated in the proposed approach. Hence, the chances of the proposed approach being trapped in local minimum is very less. This leads to enhanced accuracy of the system. Also, the "backprop" algorithm usually suffers from overlearning which is also eliminated in the proposed approach. Thus, MDCNN is beneficial in terms of accuracy measures over the conventional DCNN algorithm.

TABLE 4. Performance measures of DCNN approach.

	TP	TN	FP	FN	TPR	TNR	A (%)
Metastasis	48	84	4	4	0.92	0.95	94.2
Meningioma	27	107	3	3	0.9	0.97	95.7
Glioma	26	106	4	4	0.86	0.96	94.2
Astrocytoma	24	108	4	4	0.85	0.96	94.2
Average Results					0.88	0.96	94.5

A=Accuracy

TABLE 5. Performance measures of MDCNN approach.

	TP	TN	FP	FN	TPR	TNR	A (%)
Metastasis	49	86	2	3	0.94	0.97	96.4
Meningioma	28	108	2	2	0.93	0.98	97.1
Glioma	28	108	2	2	0.93	0.98	97.1
Astrocytoma	25	108	4	3	0.89	0.96	95
Average Results					0.92	0.97	96.4

A=Accuracy

B. RECIEVER OPERATING CURVES OF CLASSIFIERS

The performance of the classifiers is also analyzed in terms of ROC curves. ROC curves usually provide a plot between the sensitivity and specificity criterion. These curves are drawn by considering 4 different threshold points (0.2, 0.4, 0.6 and i0.8). These threshold values are predefined values

used to differentiate the correct classification and misclassification in the output layer neurons. For example, if a threshold value is chosen as 0.2, then a training image from meningioma category is said to be correctly classified only if the output value of that specific predefined neuron shows a value between 0 and 0.2. The process is repeated for all the images with the different threshold points. The sensitivity and specificity values are then estimated at these threshold points. The ROC curves are shown in Figure 6.

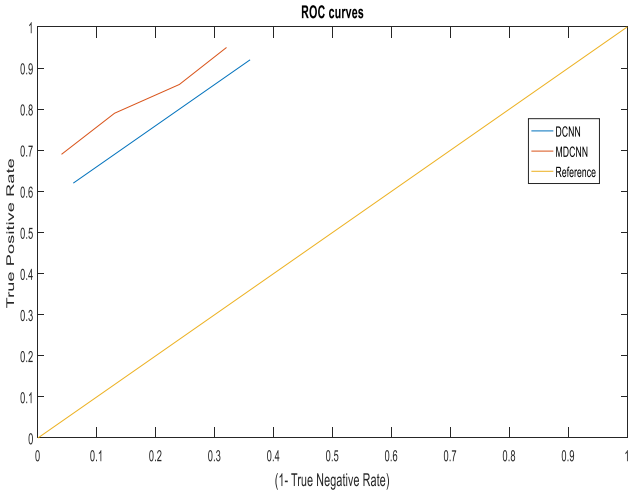

FIGURE 6. ROC curves of the classifiers.

The centre line denotes the reference line. Any classifier whose curves fall below the reference line is a weak classifier. The curves of the best classifier always lie above the reference line. In addition, the classifier with the curve close to the top left corner is the best classifier. The top leftmost corner denotes the best value of sensitivity and specificity. From Figure 6, it is evident that the performance of the MDCNN classifier is better than the conventional DCNN. Thus, irrespective of the threshold points, MDCNN guarantee better accuracy than the conventional DCNN. This analysis also proves that the proposed approach is efficient in terms of both sensitivity (true positive rate) and specificity (true negative rate).

C. COMPUTATIONAL COMPLEXITY OF CLASSIFIERS

The computational complexity is estimated by analyzing the number of mathematical operations required for the training of the algorithms. The training algorithm includes the weight adjustment process of the convolutional layer and the weight adjustment process of the fully connected layer. Since the weight adjustment of convolutional layer is same for both algorithms, the analysis of the weight adjustment in this layer is not carried out in this work. The difference between the two algorithms lie in the weight adjustment process of fully connected layer only. Hence, the computational complexity analysis is done only for the weight adjustment process in the fully connected layer.

1) ANALYSIS OF WEIGHT ADJUSTMENT PROCESS IN FULLY CONNECTED LAYER OF DCNN

The complete process is subdivided into small steps to determine the number of mathematical operations required for each step. The calculations of mathematical operations in the forward pass is given below:

In the first step, the number of mathematical operations required for NET value calculation of the hidden layer is estimated. If a is the number of hidden layer neurons and b is the number of output layer neurons, then the number of multiplication operations required are '$a \times b$'. The required number of addition operations for executing this step are '$(a \times b)/2$'. The number of addition operations required for OUT value estimation in the hidden layer is 'b' and the number of division operations required is also 'b'. The same process is repeated for the output layer neurons also. If c is the number of neurons in the output layer, then the number of multiplication operations required for NET value calculation is '$b \times c$' and the number of addition operations required is '$(b \times c)/2$'. The number of addition operations required for OUT value estimation in the output layer is 'c' and the number of division operations required is also 'c'. Thus, the overall operations is given by $[(a \times b) + (b \times c)] + [\{(a \times b)/2\} + \{(b \times c)/2\}] + [b + c]$.

The calculations of mathematical operations in the reverse pass is given below:

In the second step, the number of mathematical operations in the reverse pass is estimated. These calculations are based on a single hidden layer. The weight adjustment in the output layer includes 1 addition operation and 4 multiplication operations. The weight adjustment in the hidden layer includes 5 additions and 9 multiplication operations (since the number of output layer neuron is 4, the operations are increased by a factor of 4. If the number of hidden layers is increased, these operations increase drastically.

Thus, the overall complexity of the conventional fully connected layer is given by '$T\{([(a \times b) + (b \times c)] + 13) + ([\{(a \times b)/2\} + \{(b \times c)/2\}] + 6) + [b + c]\}$'. It can be noted that the computational complexity is huge for more layers and a greater number of neurons. Also, this algorithm is dependent on iterations 'T'. Hence, the complexity increases with increase in value of 'T'.

2) ANALYSIS OF WEIGHT ADJUSTMENT PROCESS IN FULLY CONNECTED LAYER OF MDCNN

In the proposed approach, the first advantage is that it is independent of iterations. Hence, the computational complexity is significantly reduced. Also, the weight estimation process is relatively simple in comparison to the weight adjustment process of the conventional approach. The weight estimation process of output layer includes ''$b \times c$' multiplications and the number of addition operations required is '$(b \times c)/2$'. The weight estimation process of hidden layer includes ''$a \times b$' multiplications and the number of addition operations required is '$(a \times b)/2$'. Thus, the overall

operations are only $[(a \times b) + (b \times c)] + [\{(a \times b)/2\} + \{(b \times c)/2\}]$. Also, there is no reverse pass as in conventional approach which further reduces the computational complexity.

Thus, the proposed approach is superior to the conventional approach in terms of computational complexity. Since the experiments are carried out on real world images, a comparative analysis with other works are not reported in this work. It may be kindly noted that the performance comparison with other literature works must be carried out under same environmental conditions including the dataset. However, an overall analysis from the literature reveals that the accuracy of CNN based medical image classification approaches are in the range of 90%-95%. The accuracy of the proposed approach is sufficiently higher than the conventional CNN based image classification approaches.

VI. CONCLUSIONS

A modified deep convolutional neural network is proposed in this work for MR brain image classification. The proposed approach is analyzed in terms of accuracy and computational complexity. An approximate improvement of 3% is achieved with the proposed approach in comparison to the conventional CNN approach. A sufficient improvement in the True Positive Rate and True Negative Rate is also seen from the experimental results. In the proposed approach, the weights are not adjusted in the fully connected layer of the proposed approach. This reduces the computational complexity to high extent which makes it suitable for practical applications. Thus, an alternate for conventional CNN is proposed in this work which performs better than the conventional CNN.

REFERENCES

[1] H. Mohsen, E.-S. A. El-Dahshan, E.-S. M. El-Horbaty, and A.-B. M. Salem, "Classification using deep learning neural networks for brain tumors," *Future Comput. Inform. J.*, vol. 3, pp. 68–71, Jun. 2018.
[2] X. W. Gao, R. Hui, and Z. Tian, "Classification of CT brain images based on deep learning networks," *Comput. Methods Programs Biomed.*, vol. 138, pp. 49–56, Jan. 2017.
[3] H. Choi and K. H. Jin, "Predicting cognitive decline with deep learning of brain metabolism and amyloid imaging," *Behavioural Brain Res.*, vol. 344, pp. 103–109, May 2018.
[4] E. Gibson *et al.*, "NiftyNet: A deep-learning platform for medical imaging," *Comput. Methods Programs Biomed.*, vol. 158, pp. 113–122, May 2018.
[5] G. Litjens *et al.*, "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42, pp. 60–88, Dec. 2017.
[6] S. Michael and F. Gregory, "Using a model of human visual perception to improve deep learning," *Neural Netw.*, vol. 104, pp. 40–49, Aug. 2018.
[7] L.-L. Zeng, "Multi-site diagnostic classification of schizophrenia using discriminant deep learning with functional connectivity MRI," *EBioMedicine*, vol. 30, pp. 74–85, Apr. 2018.
[8] C. Cao *et al.*, "Deep learning and its applications in biomedicine," *Genomics, Proteomics Bioinf.*, vol. 16, pp. 17–32, Feb. 2018.
[9] M. Yang, F. Li, L. Zhang, and Z. Zhang, "Deep learning algorithm with visual impression," *Inf. Process. Lett.*, vol. 136, pp. 1–4, Aug. 2018.
[10] Z. Gandomkar, P. C. Brennan, and C. Mello-Thoms, "MuDeRN: Multi-category classification of breast histopathological image using deep residual networks," *Artif. Intell. Med.*, vol. 88, pp. 14–24, Jun. 2018.
[11] Y. Kong, J. Gao, Y. Xu, Y. Pan, J. Wang, and J. Liu, "Classification of autism spectrum disorder by combining brain connectivity and deep neural network classifier," *Neurocomputing*, vol. 324, pp. 63–68, Jan. 2019, doi: 10.1016/j.neucom.2018.04.080.
[12] H. Li, G. Li, X. Ji, and L. Shi, "Deep representation via convolutional neural network for classification of spatiotemporal event streams," *Neurocomputing*, vol. 299, pp. 1–9, Jul. 2018.
[13] A. Qayyum, S. M. Anwar, M. Awais, and M. Majid, "Medical image retrieval using deep convolutional neural network," *Neurocomputing*, vol. 266, pp. 8–20, Nov. 2017.
[14] S. Hussain, S. M. Anwar, and M. Majid, "Segmentation of glioma tumors in brain using deep convolutional neural network," *Neurocomputing*, vol. 282, pp. 248–261, Mar. 2018.
[15] U. R. Acharya, S. L. Oh, Y. Hagiwara, J. H. Tan, and H. Adeli, "Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals," *Comput. Biol. Med.*, vol. 100, pp. 270–278, Sep. 2017.
[16] J. Amin, M. Sharif, M. Yasmin, and S. L. Fernandes, "Big data analysis for brain tumor detection: Deep convolutional neural networks," *Future Gener. Comput. Syst.*, vol. 87, pp. 290–297, Oct. 2018.
[17] T. Banzato, G. B. Cherubini, M. Atzori, and A. Zotti, "Development of a deep convolutional neural network to predict grading of canine meningiomas from magnetic resonance images," *Vet. J.*, vol. 235, pp. 90–92, May 2018.
[18] J. H. Tan *et al.*, "Age-related macular degeneration detection using deep convolutional neural network," *Future Gener. Comput. Syst.*, vol. 87, pp. 127–135, Oct. 2018.
[19] J. Kim, H. Kim, S. Huh, J. Lee, and K. Choi, "Deep neural networks with weighted spikes," *Neurocomputing*, vol. 311, pp. 373–386, Oct. 2018.
[20] X. Shi, M. Sapkota, F. Xing, F. Liu, L. Cui, and L. Yang, "Pairwise based deep ranking hashing for histopathology image classification and retrieval," *Pattern Recognit.*, vol. 81, pp. 14–22, Sep. 2018.

D. JUDE HEMANTH received the B.E. degree from Bharathiar University, the M.E. degree from Anna University, and the Ph.D. degree from Karunya University, India. He has published more than 100 research papers in international journals and conferences. His cumulative impact factor is 70. His research interests include neuro imaging and computational intelligence.

J. ANITHA received the B.E. degree from Bharathiar University, the M.E. degree from Anna University, and the Ph.D. degree from Karunya University, India. She has published more than 60 papers in international journals and conferences. She has completed a research project on medical informatics sponsored by the Govt. of India. Her research interests include medical imaging and soft computing.

ANTOANELA NAAJI received the Ph.D. degree from the Politehnica University of Timisoara, Romania, in 2005. She is currently an Associate Professor of computer science with the Vasile Goldiș Western University of Arad, Romania. She has published more than 100 research papers. Her research interests include biomechanics and medical informatics.

OANA GEMAN received the Ph.D. degree in electronics and telecommunication. She is currently a Medical Bioengineer and a Post-Doctoral Researcher in computer science. She has published four books, six book chapters as co-author, and over 44 ISI articles with an IF of over 14. She has been a contributor for 10 national and international grants.

DANIELA ELENA POPESCU received the Ph.D. degree in computer science from the Politehnica University of Timisoara. She is currently a Professor with the University of Oradea and a member of the Doctoral School of the Politehnica University of Timisoara. She has published more than 50 research papers. Her areas of interests include artificial intelligence and informatics.

LE HOANG SON received the Ph.D. degree in mathematics-informatics from Vietnam National University. He has published more than 100 research papers in international journals and conferences. His major field includes soft computing and medical image processing.