Inception recurrent convolutional neural network for object recognition

Md Zahangir Alom · Mahmudul Hasan · Chris Yakopcic · Tarek M. Taha · Vijayan K. Asari

Received: 11 May 2019 / Revised: 28 October 2020 / Accepted: 12 November 2020 / Published online: 3 January 2021
© Springer-Verlag GmbH Germany, part of Springer Nature 2021

Abstract
Deep convolutional neural network (DCNN) is an influential tool for solving various problems in machine learning and computer vision. Recurrent connectivity is a very important component of visual information processing within the human brain. The idea of recurrent connectivity is rarely applied within convolutional layers, the exceptions being a couple of DCNN architectures including recurrent convolutional neural network (RCNN) in Liang and Hu (in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015) and Pinheiro and Collobert (in: ICML, 2014). On the other hand, the Inception network architecture has become popular among the computer vision community (Szegedy et al. in Inception-v4, Inception-ResNet and the impact of Residual connections on learning, 2016. arXiv:1602.07261). In this paper, we introduce a deep learning architecture called the Inception Recurrent Convolutional Neural Network (IRCNN), which utilizes the power of an Inception network combined with recurrent convolutional layers. Although the inputs are static, the recurrent property plays a huge role in modeling the contextual information for object recognition tasks and thus improves overall training and testing accuracy. In addition, this proposed architecture generalizes both Inception and RCNN models. We have empirically evaluated the recognition performance of the proposed IRCNN model using different benchmark datasets such as MNIST, CIFAR-10, CIFAR-100, and SVHN. The experimental results show higher recognition accuracy when compared to most of the popular DCNNs including the RCNN. Furthermore, we have investigated IRCNN performance against equivalent Inception networks (EIN) and equivalent Inception–Residual networks (EIRN) using the CIFAR-100 dataset. When using the augmented CIFAR-100 dataset, we achieved about 3.5%, 3.47% and 2.54% improvement in classification accuracy compared to the RCNN, EIN, and EIRN respectively. We have also conducted experiment on Tiny ImageNet-200 dataset with IRCNN, EIN, EIRN, RCNN, DenseNet in Huang et al. (Densely connected convolutional networks, 2016. arXiv:1608.06993), and DenseNet with Recurrent Convolution Layer, where the proposed model shows significantly better performance against baseline models.

Keywords IRCNN · RCNN · DCNN · Deep Learning · Object recognition

1 Introduction
In recent years, deep learning using convolutional neural networks (CNN) has shown enormous success in the field of machine learning and computer vision. CNNs provide state-of-the-art performance in various image recognition tasks including object recognition in [1–4], object detection in [5], tracking in [6], and image captioning in [7]. This technique has been applied massively in computer vision tasks such as video representation and classification in [8]. In addition, deep learning approaches are successfully used in the field of medical imaging, medical information processing, and they have achieved (near) human-level performance with the Inception v3 architecture in [9, 10]. Furthermore, the field

© Md Zahangir Alom
alomn@udayton.edu

Mahmudul Hasan
mahmud.ucr@gmail.com

Chris Yakopcic
cyakopcic@udayton.edu

Tarek M. Taha
ttaha@udayton.edu

Vijayan K. Asari
vasari1@udayton.edu

1 Department of Electrical and Computer Engineering, University of Dayton, Dayton, OH, USA

2 Comcast Labs, Washington, DC, USA
of natural language processing (NLP) and machine translation (MT) is applied deep learning techniques and shows tremendous success in this application domain [11, 12]. This technique has been used extensively in the field of speech recognition in [13]. Moreover, deep learning technique has been applied in the application domain of intelligent game development successfully in [14, 15].

Presently, deep learning-based approaches (DCNN in particular in [4]) perform very well in the domains of detection, classification, and segmentation for scene understanding. The CNN is a very powerful technique used to learn high-level and multi-scale features, which helps to extract robust and discriminative features with global contextual information within a region of an input sample. In the last few years, there are different architectures (such as AlexNet in [2], VGG in [3], NiN in [30], the All Convolutional Network in [31], GoogLeNet in [4], Inception-v4 in [26], and Residual Networks in [25, 27]) have been proposed and evaluated in different object recognition tasks. Among all the models, Inception and Residual networks are used massively for object recognition task in the field of computer vision. However, most of the hierarchical feature learning models including CNNs in [2, 4], Neocognitron in [16], and HMAX in [17] are proposed using a feed-forward architecture.

The visual cortex in the human brain consists of several visual processing units and processes information using feed-forward and feedback (recurrent connectivity) techniques in [18]. The feed-forward technique implements concurrent probabilistic inference in the visual hierarchy, whereas the feedback technique serves to integrate contextual prior or extra-classical receptive field effects in [18]. The model of the convolutional deep belief network (CDBN) is adopted this strategy, using feedback connections for propagation. This achieved very good accuracy for object classification tasks by [19]. An alternative study shows that the interaction of the early visual (V1) with various higher-order visual processing units through recurrent connections is responsible for the reintegration of information analyzed by the higher visual areas. This study also shows V1 could be used to integrate and coordinate the computation of identity (WHAT) and object location (WHERE) in the visual scene with recurrent interaction in [20]. Thus, the recurrent connectivity of synapses in the human brain plays a big role in context modeling for visual recognition tasks explained in [21, 22]. The importance of context modulation for visual recognition tasks is demonstrated in different studies in [21, 28].

However, several models are proposed based on the concept of a recurrent layer in an artificial neural network. The architecture of the general recurrent multilayer perceptron (RMLP) is used very often in the field of dynamic control in [23]. The RMLP is simply the extension of a multilayer perceptron (MLP) with recurrent connectivity in the layers [24]. Even if the input is static, the object recognition task is a dynamic process because of the presence of recurrent or top-down connections. The concept of the RMLP has been extended to include convolutional layers, resulted in the development of the RCNN [28]. The diagrams of the RMLP (left), the CNN (middle) and the RCNN (right) are shown in Fig. 1.

In this work, we have proposed a new deep learning architecture which combines two most recently proposed models: revised version of Inception network [26] and the RCNN [28]. In this proposed IRCNN model, the recurrent convolutional layers are incorporated within Inception block, and the convolution operations are performed considering different time steps. The complete IRCNN model is shown in Fig. 2. The proposed Inception block with recurrent convolution layers is shown in Fig. 3. The goal of the DCNN architecture of the Inception [26] and Residual networks [25, 27] is to implement large-scale deep networks. As the model becomes larger and deeper, the computational parameters of the architecture are increased dramatically. Thus, the model becomes more complex to train and computationally expensive. In this scenario, the recurrent property ensures better training and testing accuracy with less or equal computational parameters. For conducting experiment on larger dataset like ImageNet, the trend is to incorporate the number of layers while Dropout are used to address the over-fitting problem [26, 27]. Others research groups are trying to implement bigger and deeper DCNN architectures like Google Net [4], or a Residual network with 1001 layers [27] to achieve even better recognition accuracy. Alternatively, we are presenting an improved version of the DCNN model inspired by the information processing mechanism of the human visual cortex and the recently developed DCNN architectures such as Inception-v4 [26] and RCNN [28]. Therefore, we call this model the Inception Recurrent Convolutional

![Fig. 1 Diagram for recurrent multilayer perceptron (RMLP) in the left, convolutional neural network (CNN) in the middle and recurrent convolutional neural network (RCNN) in the right](Image 309x562 to 541x735)
Inception recurrent convolutional neural network for object recognition

Fig. 2 Overall operational flow diagram of the proposed Inception Recurrent Convolutional Neural Network (IRCNN), which consists of an IRCNN block, a transition block, and a softmax layer.

Fig. 3 Inception-Recurrent Convolutional Neural Network (IRCNN) block with different convolutional layers respect to different sizes of kernels. The upper part of the figure represents the Inception layers for different kernels and pooling operation with recurrent layers and lower part shows the internal details of recurrent convolutional operation respect to time (t = 3). Finally, the outputs are concatenated for the immediate next layer.

Neural Network (IRCNN). This model not only ensures better recognition accuracy with same number of computational parameters against the state-of-the-art DCNN architectures, but also helps to improve the overall training process of the deep learning approach. The contributions of this work are as follows:

- A new deep learning model called IRCNN is proposed with the combination of the recently developed Inception-v4 [26] and RCNN [28].
- Experimental analysis of the proposed learning model’s performance against different DCNN architectures on different benchmark datasets such as MNIST, CIFAR-10, CIFAR-100, SVHN, and Tiny ImageNet-200.
- Empirical evaluation of the impact of the recurrent convolutional layer (RCL) in the Inception Network and Residual Network (ResNet).
- Empirical investigation of the impact of RCLs on Densely connected neural networks namely DenseNet.

The remaining content is organized as follows. Section 2 reviews relevant related work. Section 3 describes the overall architecture of IRCNN. Section 4 presents the experimental results and analysis. Finally, Sect. 5 concludes the paper.

2 Related work

The related works of object recognition using convolutional neural network (CNNs) and CNNs with recurrent approach are discussed in the following sections.
2.1 CNNs for object recognition

The first self-organizing neural network model called the Neocognitron was proposed by Fukushima in 1982 [16]. Although the deep learning revolution began in 1998 with LeNet in [29], from then on, several architectures have been proposed that have shown massive success using different benchmark datasets including MNIST, SVHN, CIFAR-10, CIFAR-100, ImageNet, and many more. Of the DCNN architectures, AlexNet [2], VGG [3], NiN [30], the All Convolutional Network [31], GoogLeNet [4], Inception-v4 [26], and Residual Networks [25, 27] can be considered the most popular architectures due to their improved performance on different benchmarks for object classification. In 2012, Alex Krizhevsky et al. proposed an improved deeper version of a CNN compared to LeNet [29] and won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012. This was a significant breakthrough in the field of machine learning and computer vision, as this was the first time a deep network outperformed the alternative approaches for visual recognition tasks [2]. GoogLeNet, or Inception-v1 [4], and the Residual network [25, 28] won ILSVRC in 2014 and 2015, respectively.

Inception architecture has become very popular in the deep learning and computer vision community, and it has been refined in different ways. An Inception network with batch normalization [60] (Inception-v2) was proposed by Ioffe et al. The Inception network (Inception-v3) was proposed with factorization ideas in [35]. In most cases, the improvement in deep learning approaches has been due to the development of the following components: initialization techniques of DCNNs [32], new deep network architectures [33, 34], optimization of deep network structures (depending upon computational parameters) [36], deeper and wider deep networks [37, 38], activation functions for deep learning approaches [39], and optimization methods for training DCNNs [40, 41]. Some researchers have been focused on design alternatives that produce the same level of recognition accuracy as state-of-the-art architectures (like Inception-V4 with Residual Net [26]) with fewer computational parameters [35]. In this work, we have emphasized the development of an alternative DCNN architecture called the IRCNN.

2.2 CNN with recurrent layers for object recognition

Presently, most researches have been focused on improving recognition accuracy with new DCNN models. Very little research has been conducted on recurrent architectures within the layers of a CNN. Recurrent connectivity is inspired by the human visual cortex, and its importance is demonstrated for object recognition tasks using 100 different object categories containing naturally occurring variations such as location, rotation, size, and lighting [42]. In addition, in a real-world scenario, occlusion and low contrast have a big impact on visual recognition tasks. The recurrent network promotes robust object recognition in this particular situation [43].

As far as recurrent connectivity in DCNNs is concerned, the relationship between the Residual network (ResNet) [27], RNNs, and the visual cortex shows that a shallow RNN with weight sharing among the layers is exactly equivalent to a very deep ResNet. The study shows that the RNNs provide better recognition accuracy than ResNet while having an order of magnitude fewer parameters [44]. In 2015, Ming et al. proposed the first RCNN structure tested using object recognition tasks. The architecture consists of several blocks of recurrent convolutional layers followed by a max-pooling layer. In the second to last layer of the structure, global max-pooling is used followed by a softmax layer at the end. In 2015, this architecture reported state-of-the-art accuracy for object classification on different benchmarks [28]. Another RCNN-based approach was proposed for scene labeling for large input context modeling with limited capacity networks, and it achieved state-of-the-art performance on different scene understanding datasets [45]. The long-term recurrent convolutional network (LRCN) was proposed for visual recognition and description by Donahue et al. [46]. This architecture uses a combination of two popular techniques, CNN and LSTM. The features are extracted through the CNN, and LSTM is applied to identify how features vary with respect to time. This model shows outstanding performance for visual description [46].

From the above discussion, it can be concluded that DCNN architectures with improved techniques show enormous achievement when performing visual recognition tasks. The following section demonstrates the theoretical details of proposed deep IRCNN learning architecture.

3 Inception Recurrent Convolutional Neural Networks (IRCNNs)

The proposed architecture (IRCNN) is based on several recently developed deep learning architectures, including Inception Nets [26, 35] and RCNNs [28]. It tries to reduce the number of computational parameters, while providing better recognition accuracy. As shown in Fig. 2, the IRCNN architecture consists of general convolution layers, IRCNN blocks, transition blocks, and a softmax layer at the end. One of the most novel features of this work is the introduction of recurrence into the Inception module, as shown in the IRCNN block in Fig. 3. The key feature of Inception-v4 is that it concatenates the outputs of multiple differently sized convolutional kernels in the Inception block [26]. Inception-v4 is a simplified version of Inception-v3, using lower rank filters and pooling layers. Inception-v4, however, combines
Residual concepts with Inception networks to improve the overall accuracy over Inception-v3. The outputs of Inception layers are added with the inputs to the Inception–Residual module. In this work, we utilize the Inception concepts from Inception-v4 [26].

3.1 IRCNN block

The IRCNN block performs recurrent convolution operations with different sized kernels (see Fig. 3). In the recurrent structure, the inputs to the next time step are the sum of the convolutional outputs of the present time step and previous time steps. The same operations are repeated based on the number of time steps that are considered. As the input and output dimensions do not change, this is simply an accumulation of feature maps with respect to the time step are considered. This helps to strengthen the extraction of the target features. As shown in Fig. 3, one of the paths of Inception block contains an average pooling operation is applied before the recurrent convolution layer. This particular pooling layer, a 3 × 3 average pooling with stride 1 × 1 is applied by keeping the border size same, results in output samples with the same dimensions as the inputs. The overlapping average pooling technique helps in the regularization of the network [2].

The operations of each recurrent convolution layer (RCL) in the IRCNN block are similar to operations as mentioned in [28]. To describe these operations, consider a vectorized patch centered at (i, j) of an input sample x_l for the Kth feature map in the RCL unit. The O_{ijk}^l refers the output of lth layer at time step t. The output can be represented as:

\[ O_{ijk}^l(t) = (w_{ik}^f)^T \odot x_{l}^{f(i,j)}(t) + (w_{ik}^r)^T \odot x_{l}^{r(i,j)}(t-1) + b_{k} \]

Here \( x_{l}^{f(i,j)} \) and \( x_{l}^{r(i,j)}(t-1) \) are the inputs for a standard convolutional layer and an RCL, respectively. The \( w_{ik}^f \) and \( w_{ik}^r \) are the weights for the standard convolutional layer and the RCL of Kth feature map, respectively, while \( b_{k} \) is the bias. Then, the outputs of RCL pass through an activation function. Thus, the final output for the layer at time step t is:

\[ y_{ijk}^l(t) = f(O_{ijk}^l(t)) = \max(0, O_{ijk}^l(t)) \]

where f is the standard Rectified Linear Unit (ReLU) activation function. The Local Response Normalization (LRN) function is applied on the outputs of each kernel in the IRCNN block [2]:

\[ z = \text{norm}(y_{ijk}^l(t)) \]

The outputs of the IRCNN block with respect to the different kernel sizes 1 × 1 and 3 × 3 and average pooling operations followed by 1 × 1 are defined as \( z_{1\times1}(x) \), \( z_{3\times3}(x) \), and \( z_{1\times1}(x) \), respectively. The final output \( z_{\text{out}} \) of the IRCNN block can be expressed as:

\[ z_{\text{out}} = z_{1\times1}(x) \odot z_{3\times3}(x) \odot z_{1\times1}(x) \]

Here \( \odot \) represents the concatenation operation with respect to the channel axis of the output samples. In this implementation, we have used \( r = 3 \) that indicates the four recurrent convolutional operation have been performed in each IRCNN block (individual path) which is clearly shown in Fig. 3. The outputs of the IRCNN block become the inputs that are fed into the transition layer.

3.2 Transition block

In the transition block, three operations (convolution, pooling, and Dropout) are performed depending upon the placement of the block in the network. According to Fig. 2, we have applied all of the operations in the very first transition block, whereas in the second transition block, we have only used convolution with dropout operations. The third transition block consists of convolution, global-average pooling, and drop-out layers. The global-average pooling layer is used as an alternative to a fully connected layer. There are several advantages of a global-average pooling layer. First, it is very close in operation to convolution, hence enforcing correspondence between feature maps and categories. The feature maps can be easily interpreted as class confidence. Second, it does not need computational parameters, thus helping to avoid over-fitting of the network. Late use of the pooling layer is advantageous because it increases the number of non-linear hidden layers in the network. Therefore, we have applied only two special pooling layers in the first and third transition block of this architecture. Special pooling is carried out with the max-pooling layer 3 × 3 in this network. (Not all transition blocks have pooling layer.) The max-pooling layers perform operations with a 3 × 3 patch and a 2 × 2 stride over the input samples. Since the non-overlapping max-pooling operation has a negative impact on model regularization, we used overlapped max-pooling for regularizing the network. This is very important for training a deep network architecture [2]. Eventually, a global-average pooling layer is used as an alternative of fully connected layers. Finally, the softmax logistic regression layer is used at the end of the IRCNN architecture.

3.3 Optimization of network parameters

To keep the number of computational parameters low compared to other traditional DCNN approaches like AlexNet [2] and VGGNet [3], we have used only 1 × 1 and 3 × 3 convolutional filters in this implementation (inspired by the NiN [30]
and SqueezeNet [36] models). There are significant benefits to using smaller sized kernels, such the ability to incorporate more nonlinearity in the network. For example, we can use a stack of two $3 \times 3$ respective fields (without placing any pooling layer in between) as a replacement for one $5 \times 5$ and a stack of three $3 \times 3$ respective fields instead of a $7 \times 7$ kernel size according to [3]. The benefit of adding a $1 \times 1$ filter is that it helps to increase the nonlinearity of the decision function without having any impact on the convolution layer. Since the size of the input and output features do not change in the IRCNN blocks, it is just a linear projection on the same dimension with nonlinearity added using a ReLU. We have used a dropout of 0.5 after each convolutional layer in the IRCNN block.

Finally, we have used a softmax or a normalized exponential function in [47] layer at the end of the architecture. For an input sample $x$ and a weight vector $w$, and $K$ distinct linear functions, the softmax operation can be defined for $i$th class as follows:

$$P(y = i|x) = \frac{e^{x^T w_i}}{\sum_{k=1}^{K} e^{x^T w_k}}$$

### 3.4 Network architectures

We have conducted experiments with different models including RCNN in [28], EIN which is same as Inception-v3 network in [26, 35], EIRN which is a small-scale implementation of Inception-v4 model in [26], and the proposed IRCNN. These models are evaluated with different numbers of convolutional layers in the convolutional blocks while the number of layers is determined with respect to time step $t$. In these implementations, we have used $t = 2$ that refers the RCL block contains one forward convolution followed by two RCLs, and $t = 3$ is for one forward convolution with three RCLs. To experiment with MNIST dataset, we have used a model consists of two forward convolutional layers at the beginning, two IRCNN blocks followed by transition blocks, and softmax layer at the end. In the case of CIFAR-10, CIFAR-100, and SVHN datasets, we have used an architecture with two convolutional layers at the beginning, three IRCNN blocks followed by transition blocks, a dense layer, and a softmax layer at the end of the model. For this model, we have considered 16 and 32 feature maps for the first three convolutional layers, and 64, 128, and 256 feature maps are used in first, second, and third IRCNN blocks, respectively. The overall model diagram is shown in Fig. 2. This model contains around 3.2 million (M) network parameters.

We have used a model with four IRCNN blocks followed by transition layers, fully connected layer, and softmax layer for the experiment on Tiny ImageNet-200 dataset. In addition, we have almost doubled the number of feature maps in each of the forward convolution layers and RCLs in IRCNN blocks compared to the model that is used for CIFAR-100, which significantly increases the number of network parameters to approximately 9.3 M. However, EIN and EIRN models are implemented with the same structure of IRCNN model with Inception and Inception-Residual modules, respectively. For conducting experiment on Tiny ImageNet-200 dataset, batch normalization (BN) is used instead of LRN in IRCNN, RCNN, EIN, and EIRN models. We have skipped Eq. 3, and the concatenation operation is performed directly on the output of Eq. 2. In this case, BN is applied at the end of IRCNN block on $z_{out}$. Furthermore, we have empirically investigated the impact of RCLs on the DenseNet model as in [63]. In this implementation, we have incorporated the RCL layers (where $t = 2$ refers one forward convolution and two RCLs) as a replacement for forward convolutional layers within the dense block. An BN layer is used in the dense block with RCLs. We only used four dense blocks, with four layers in each block and a growth rate 6 according to [63]. The experimental result shows significant improvement in training, validation, testing accuracies with DenseNet with RCLs against original DenseNet model in [63].

### 4 Experiments

We have evaluated the proposed IRCNN method with a set of experiments on different benchmark datasets: MNIST [48], Cifar-10[49.], Cifar-100 [49], SVHN [50], and Tiny ImageNet dataset [64] and compared against different models. The entire experiment has been conducted on Linux environment with Keras [61] and Theano [62] in the Backend running on the single GPU machine with NVIDIA GEFORCE GTX-980 Ti.

#### 4.1 Training Methodology

In the first experiment, we have trained the proposed IRCNN architecture using the stochastic gradient descent (SGD) optimization function with the default initialization technique for deep networks found in Keras [61]. We set the Nesterov momentum to 0.9 [51] and decay to 9.99e-07 9.99 × 10−7. Second, we performed experiment with our proposed approach with the layer-sequential unit-variance (LSUV) technique, which is a simple method for the weights initialization in a deep neural network [32]. We have also used a very recently proposed improved version of the optimization function based on “Adam” known as EVE [40]. The following parameters are used for the EVE optimization function: The value of the learning rate ($\lambda$) is 1e-4, decay ($\gamma$) is 1e-4, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\beta_3 = 0.999$ b, $\kappa = 0.1$, $K = 10$, and $\epsilon = 1e - 08$. The $\beta_1, \beta_2 \in [0, 1]$ are exponential decay
Table 1  Testing errors (%) of IRCNN on MNIST, CIFAR-10(C-10), CIFAR-100(C-100), and SVHN. Here “+” indicates standard data augmentation using random horizontal flipping. IRCNN achieves lower testing in most of the cases indicates with bold

| Methods            | # network parameters | MNIST | C10   | C10*  | C100 | C100* | SVHN* |
|-------------------|----------------------|-------|-------|-------|------|-------|-------|
| Maxout [52]       | > 5 M                | 0.45  | 11.6  | 9.38  | –    | 38.57 | 2.47  |
| NIN [30]          | ~ 1.1 M              | 0.47  | 10.41 | 8.81  | 35.68| –     | 2.35  |
| DSN [55]          | ~ 1.0 M              | 0.39  | 9.69  | 7.97  | –    | 34.57 | 1.93  |
| Prob maxout [57]  | > 5 M                | –     | 9.39  | –     | –    | 38.14 | 2.39  |
| ALL-CNN [31]      | 1.3 M                | –     | 9.08  | 7.25  | –    | 33.71 | –     |
| Highway Network   | ~ 1.4 M              | –     | –     | 7.72  | –    | 32.24 | –     |
| RCNN [28]         | 1.86 M               | 0.31  | 8.69  | 7.09  | –    | 31.75 | 1.77  |
| dasNet [54]       | –                    | –     | –     | 9.22  | –    | 33.78 | –     |
| FitNet [58]       | –                    | –     | –     | 8.39  | –    | 35.04 | –     |
| DropConnect (5 Nets) [53] | –     | 1.12  | –     | 9.41  | –    | –     | 1.94  |
| CNN + Tree [59]   | –                    | –     | –     | –     | –    | 36.85 | –     |
| IRCNN +SDG        | 3.2 M                | 0.32  | 8.41  | 7.37  | 34.13| 31.22 | 1.89  |
| IRCNN + LSUV + EVE | 3.2 M               | **0.29** | **8.17** | **7.11** | **30.87** | **28.24** | **1.74** |

rates for moment estimation in Adam. The \( \beta_3 \in [0, 1] \) is an exponential decay rate for computing relative changes. The \( \kappa \) and \( K \) values are lower and upper thresholds for relative change and \( \epsilon \) is a fuzzy factor [40]. It should be noted that we used the \( l_2 \) – norm with a valued of 0.002 for weight regularization.

In both experiments, we have used ReLU activation functions. We have generalized the network with dropout (0.5). Only horizontal flipping technique was applied when performing data augmentation. During the training of MNIST and SVHN, we have used 200 epochs with a mini-batch size of 128. We train the models for 350 epochs with a 128 batch size for CIFAR-10 and 100. For the impartial comparison, we have trained and tested against equivalent Inception networks and Inception–Residual networks. By equivalent, we mean having the same number of layers and computational parameters.

5 Results

5.1 Mnist

MNIST is one of the most popular datasets for handwritten digits from 0 to 9 [36], the dataset contains 28 \( \times \) 28 pixels grayscale images with 60,000 training examples and 10,000 testing examples. For this experiment, we trained the proposed model with two IRCNN blocks (IRCNN block 1 and IRCNN block 2) and used the ReLU activation function. The model was trained with 60,000 samples, and 10,000 samples are used for validation set. Eventually, the trained network was tested with 10,000 testing examples. We obtained a test error of 0.32% with the IRCNN and the SGD and achieved about 0.29% error for the IRCNN when initializing with LSUV [32] and the EVE [40] optimization function. This is the best accuracy compared to the RCNN, as well as the other state-of-the-art networks. The summary of the classification accuracies is given in Table 1. No data augmentation techniques have been applied in this experiment on MNIST. On the contrary, global contract normalization and ZCA whitening were applied in the experiments using most of the mentioned models in Table 1.

5.1.1 Cifar-10

CIFAR-10 is an object classification benchmark [37] consisting of 32 \( \times \) 32 color images representing ten classes. It is split into 50,000 samples for training and 10,000 samples for testing. The experiment was conducted with and without data augmentation. The entire experiment was conducted on models similar to the one shown in Fig. 2. Using the proposed approach, we achieved about 8.41% error without data augmentation and 7.37% error with data augmentation using the SGD technique. These results are better than those of most of the recognized DCNN models stated in Table 1.

Better performance is observed from the IRCNN with LSUV [32] as the initialization approach and EVE [40] as the optimization technique. The results show around 8.17% and 7.11% error without and with data augmentation, respectively. When comparing these results to those of the different models in Table 1, it can be observed that our proposed approach provides better accuracy compared to various advanced and hybrid models. The training and validation loss of the experiment on CIFAR-10 of this proposed model are shown in Fig. 4. Figure 5 shows training and validation accuracy of IRCNN with SGD and LSUV + EVE.
5.1.2 Cifar-100

This is another benchmark for object classification from the same group [49]. The dataset contains 60,000 (50,000 for training and for 10,000 testing) color 32 × 32 images, and it has 100 classes. We used SGD and LSUV [32] as the initialization approach with the EVE optimization technique [40] in this experiment. The experimental results are shown in Table 1. In both cases, the proposed technique shows state-of-the-art accuracy compared with different DCNN models.

IRCNN + SGD shows about 34.13% classification errors without data augmentation and 31.22% classification errors with data augmentation. In addition, this model achieved around 30.87% and only 28.24% errors with SGD and LSUV + EVE on augmented dataset. This is the highest accuracy achieved in any of the deep learning models summarized in Table 1. For augmented datasets, we have achieved 71.76% recognition accuracy with LSUV + EVE, which is about a 3.51% improvement compared to RCNN [28]. Figure 6 shows the training and validation loss of the IRCNN for both experiments using the CIFAR-100 dataset with data augmentation (with and without initialization and optimization). In the first experiment, we used the IRCNN with a LSUV initialization approach and the EVE optimization function. The default initialization approach of Keras and the SGD optimization method are used in the second experiment. It is clearly shown that the proposed model has lower error in the both experiments, showing the effectiveness of the proposed IRCNN learning model. The training and testing accuracy of the IRCNN with LSUV and EVE are shown in Fig. 7.
5.1.3 Street view house numbers (SVHN)

SVHN (Netzer et al. 2011) is one of the most challenging datasets for street view house number recognition [50]. This dataset contains color images representing house numbers from Google Street View. In this experiment, we have considered the second version, which consists with $32 \times 32$ color examples. There are 73,257 samples in the training set and 26,032 samples in testing set. In addition, this dataset has 531,131 extra samples that are used for training purposes. As single input samples of this dataset contain multiple digits, the main goal is to classify the central digit. Due to the huge variation in color and brightness, this dataset is much more difficult to classify compared to the MNIST dataset. In this case, we have experimented with the same model as is used in CIFAR-10 and CIFAR-100. We have used the same preprocessing steps applied in the experiments of RCNN [28]. The experimental results show better recognition accuracy, as shown in Table 1. We have obtained around 1.89% testing errors with IRCNN + SGD and 1.74% errors with IRCNN + LSUV + EVE, respectively. It is noted that local contrast normalization (LCN) is applied during experiments of Max-Out [52], NiN [30], DSN [55], and Drop Connect [53]. The reported results of CNN with drop connection are based on the average performance of five networks [53].

5.2 Impact of recurrent convolution layers

The proposed architecture also performs well when compared to other recently proposed optimized architectures. A DCNN architecture called FitNet4 has conducted experiment with LSUV initialization approach, and it only achieved 70.04% classification accuracy with data augmentation using mirroring and random shifts for CIFAR-100 [32]. On the other hand, we have only applied random horizontal flipping for data augmentation in this implementation and achieved about 1.72% better recognition accuracy against FitNet4 [57]. For an impartial comparison with the EIN and EIRN models (same architecture but smaller version of Inception-v3 and Inception-v4 models), we have implemented the Inception network with the same number of layers and parameters as in the transition and Inception block. Instead of using recurrent connectivity in the convolutional layers, we have just used sequential convolutional layers for the same time step with the same kernels. During the implementation of EIRN, we only added Residual connection in the Inception–Residual block, where the inputs of the Inception–Residual block are added with the outputs of that particular block. The batch normalization (BN) is used in the IRCNN, EIN, and EIRN models.

In this case, all of the experiments have been conducted on the augmented CIFAR-100 dataset in [49]. The model loss and accuracy for both training and validation phases are shown in Figs. 8 and 9, respectively. From both figures, it can be clearly observed that this proposed model shows lower loss and the highest recognition accuracy during validation phase compared with EIN and EIRN, proving the effectiveness of the proposed model. It also demonstrates the advantage of recurrent layers in Inception networks. The testing accuracy of IRCNN, EIN, and EIRN on CIFAR-100 dataset is shown in Fig. 10. It can be summarized that our proposed model of IRCNN shows around 3.47% and 2.54% better testing accuracy compared to EIN and EIRN, respectively.

5.3 Evaluation on Tiny ImageNet-200 dataset

In this experiment, we have evaluated the proposed technique on the Tiny ImageNet-200 dataset. This dataset contains 100,000 samples for training, 10,000 samples for validation, and 10,000 samples for testing [64]. These images are sourced from 200 different classes of objects. The key differ-
ence between the main ImageNet dataset and Tiny ImageNet is that the images are down-sampled from $224 \times 224$ to $64 \times 64$. The main impact of down-sampling is a loss of detail. Therefore, down-sampling the images might lead the ambiguity problem which may have an effect on overall model accuracy. The original ImageNet image size is $482 \times 418$ pixels, where the average object scale is 17%. The size of the images in this experiment is $64 \times 64$, which makes the Tiny ImageNet problem even harder. Some of the example images are shown in Fig. 11.

We have experimented with IRCNN, EIN, EIRN, and RCNN models with almost same number of parameters shown in Table 2. The SGD with a starting learning rate of 0.001, a batch size 64, and a total number of 75 epochs were used. In this case, we used the transfer learning approach where weights have been stored after every 25 epochs, and then reused as initial weights for the next 25 epochs. The learning rate is decreased by the factor of 10, and weight decay is decreased with respect to the number of epochs of 25 for single time evaluation. The impact of transfer learning is clearly observed during the validation accuracy of IRCNN, EIRN, EIN, and RCNN which is shown in Figs. 12 and 13.

In the testing phase, we have evaluated the proposed approaches for Top-1% and Top-5% testing accuracy. Table 2 shows the testing accuracy for all the models including RCNN and DenseNet. According to Table 2, the IRCNN provides better performance compared to EIN, EIRN, and

---

Table 2: Top-1% and Top-5% testing accuracy on Tiny ImageNet-200 dataset

| Methods     | Parameters | Top-1% | Top-5% |
|-------------|------------|--------|--------|
| EIN         | 9.3 M      | 45.27  | 67.73  |
| RCNN        | 9.5 M      | 47.36  | 69.17  |
| EIRN        | 9.3 M      | 51.14  | 71.41  |
| IRCNN       | 9.3 M      | 51.92  | 72.04  |
| DenseNet    | 1.0 M      | 50.85  | 70.22  |
| DenseNet-RCL| 1.0 M      | 51.23  | 70.57  |

---

Fig. 10: Testing accuracy of proposed IRCNN model against EIN and EIRN on augmented CIFAR-100 dataset

Fig. 11: Some sample images from Tiny ImageNet dataset

Fig. 12: Validation accuracy for IRCNN, EIN, EIRN, and RCNN on the Tiny ImageNet 200 dataset

Fig. 13: Validation accuracy of DenseNet and DenseNet with a recurrent convolutional layer (RCL)
RCNN with almost same number of parameters for object recognition task on the Tiny ImageNet-200 dataset.

We have also conducted experiments with DenseNet [63] and DenseNet with RCL on the Tiny ImageNet-200 dataset. The experimental results show that DenseNet with RCLs provides about 0.38% improvement in Top-1% accuracy compared to DenseNet with only ~ 1 M network parameters. The experimental results show DenseNet with RCLs providing higher testing accuracy in both Top-1% and Top-5% compared against DenseNet model [63].

### 5.4 Evaluation

From the above empirical evaluations, it can be concluded that this proposed IRCNN architecture provides better recognition accuracy compared to different deep learning models in most of the cases, demonstrating the robustness of the proposed deep learning model. This model also shows better recognition performance with the same number of computational parameters 3.12 M against the EIN and EIRN models. Furthermore, if we observe the figures for model loss and accuracy, it can be clearly seen that this proposed model demonstrates less loss with better recognition accuracy. We have also empirically evaluated the rate of convergence of our proposed IRCNN algorithm compared with traditional EIN and EIRN models. The proposed model converged earlier with much lower model loss compared to EIN and EIRN. Furthermore, we have tested our proposed approaches on the Tiny ImageNet dataset in Sect. 4.4. From the experimental results, it is clearly observed that the IRCNN provides around 0.78 better recognition Top-1% accuracy compared to EIRN which is shown in Table 2. We have also evaluated the impact of RCL with the DenseNet model where DenseNet with RCL in the dense block achieves better performance when compared to DenseNet with equivalent depth and number of parameters.

### 5.5 Computational time

The computational cost (in seconds) per epoch of IRCNN, RCNN, EIN, EIRN, DenseNet, and DenseNet with RCLs models for different benchmark datasets is provided in Table 3. From this table, it can be seen that as the model becomes bigger, it takes more time per epoch. However, DenseNet takes significantly higher time per epoch compared to other models even though the number of network parameters of this model is less.

### 5.6 Introspection

In this work, we have augmented data applying only random horizontal flipping techniques whereas other models published the results with more data augmentation with transition, central crop, and ZCA. This proposed model will provide further better recognition accuracy when using datasets with additional augmentation techniques.

### 6 Conclusion

In this paper, we have proposed a new architecture: Inception Recurrent Convolutional Neural Network (IRCNN) for object recognition where we have utilized the power of recurrent techniques for context modulation with the architecture of Inception networks. The experimental results show the promising recognition accuracy compared with different state-of-the-art deep convolutional neural networks (DCNN) models on different benchmark datasets such as MNIST, CIFAR-10, CIFAR-100, and SVHN. However, when the proposed IRCNN architecture is initialized with LSUV initialization technique, and optimization function of EVE, it achieved an object recognition accuracy of 71.76% on the CIFAR-100 dataset. This is about a 3.5% improvement with respect to RCNN [28]. In addition, we empirically investigated our model on the Tiny ImageNet dataset and demonstrated that proposed architecture with RCLs outperforms both the equivalent Inception network and the Inception–Residual Network models. Furthermore, this observation is also true in case of DenseNet and DenseNet with RCLs which is experimentally investigated on the Tiny ImageNet-200 dataset. Moreover, this proposed architecture accelerates the training procedure with faster convergence, which is a big concerning issue right now for training large-scale deep learning approach.

### Acknowledgements

I would like to thank for providing Tiny ImageNet-200 dataset, which is the default course project of CS231N at Stanford University (https://tiny-imagenet.herokuapp.com/).
References

1. Schmidhuber, Jürgen: Deep learning in neural networks: an overview. Neural Netw. 61, 85–117 (2015)
2. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Adv. Neural. Inf. Process. Syst. 60(6), 84–90 (2012)
3. Simonvan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 (2014)
4. Szegedy, C. et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015)
5. Girshick, R. et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014)
6. Wang, N. et al.: Transferring rich feature hierarchies for robust visual tracking. arXiv:1501.04387 (2015)
7. Mao, J. et al.: Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv:1412.6632 (2014)
8. Ballas, N. et al.: Delving deeper into convolutional networks for learning video representations. arXiv:1511.06432 (2015)
9. Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M., Thrun, S.: Dermatologist-level classification of skin cancer with deep neural networks. Nature 542(7639), 115–118 (2017)
10. Gulshan, V., Peng, L., Coram, M., Stumpe, M.C., Wu, D., Narayanaswamy, A., Kim, R.: Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. JAMA 316(22), 2402–2410 (2016)
11. Collobert, R., Jason, W.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning. ACM, (2008)
12. Manning, C.D. et al.: The Stanford CoreNLP Natural Language Processing Toolkit. ACL (System Demonstrations), (2014)
13. Hinton, G., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Magazine 29(6), 82–97 (2012)
14. Mnih, V. et al.: Playing atari with deep reinforcement learning. arXiv:1312.5602 (2013)
15. Lillicrap, T.P. et al.: Continuous control with deep reinforcement learning. arXiv:1509.02971 (2015)
16. Fukushima, K., Miyake, S.: Neocognitron: a self-organizing neural network model for a mechanism of visual pattern recognition. In: Competition and Cooperation in Neural Nets (pp. 267–362). MIT Press, (1986)
17. He, K. et al.: Identity mappings in deep residual networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015)
18. LeCun, Y., et al.: Gradient-based learning applied to document recognition. Proc. IEEE 86(11), 2278–2324 (1998)
19. Lin, M., Qi, Ch., Shuicheng, Y.: Network in network. arXiv:1312.4400 (2013)
20. Springenberg, J.T. et al.: Striving for simplicity: the all convolutional net. arXiv:1412.6806 (2014)
21. Mishkin, D., Jiri, M.: All you need is a good init. arXiv:1511.06422 (2015)
22. Chen, T., Goodfellow, I., Shlens, J (2015) Net2net: accelerating learning via knowledge transfer. arXiv:1511.05641 (2015)
23. Shankar, S. et al.: Refining architectures of deep convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2016)
24. Szegedy, C. et al.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2016)
25. Iandola, F.N. et al.: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. arXiv:1609.02681 (2016)
26. Zagoruyko, S., Nikos, K.: Wide residual networks. arXiv:1605.07146 (2016)
27. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elu). arXiv:1511.07289 (2015)
28. Koushik, J., Hiroaki, H.: Improving stochastic gradient descent with feedback. arXiv:1611.01505 (2016)
29. Nguyen, T., et al.: Convolutional neural networks for object recognition. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), (2011)
30. O’Reilly, R.C., Wyatte, D., Herd, S., Mingus, B., Jilk, D.J.: Recurrent processing during object recognition. Front. Psychol. 4, 124 (2013)
31. Wyatte, D., Curran, T., O’Reilly, R.: The limits of feedforward vision: recurrent processing promotes robust object recognition when objects are degraded. J. Cognit. Neurosci. 24(11), 2248–2261 (2012)
32. Liao, Q., Poggio, T.: Bridging the gaps between residual learning, recurrent neural networks and visual cortex. arXiv:1604.03640, (2016)
33. Sinhwij, C., Ronan, C.: Recurrent convolutional neural networks for scene labeling. In: ICML, (2014)
34. Donahue, J. et al.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015)
35. Bishop, C.M.: Pattern recognition. Mach. Learn. 128, 1–58 (2006)
36. Kussul, Ernst, Baidyuk, Tatiana: Improved method of handwritten digit recognition tested on MNIST database. Image Vis. Comput. 22(12), 971–981 (2004)
49. Alex, K.: Learning multiple layers of features from tiny images. Technical Report, (2009)
50. Netzer, Y. et al.: Reading digits in natural images with unsupervised feature learning. (2011)
51. Sutskever, I., et al.: On the importance of initialization and momentum in deep learning. ICML 3(28), 1139–1147 (2013)
52. Goodfellow, I.J., et al.: Maxout Networks. ICML 3(28), 1319–1327 (2013)
53. Wan, L., et al.: Regularization of neural networks using dropconnect. In: Proceedings of the 30th International Conference on Machine Learning (ICML-13), (2013)
54. Stollenga, M.F. et al.: Deep networks with internal selective attention through feedback connections. Advances in Neural Information Processing Systems, (2014)
55. Lee, C.Y. et al.: Deeply-Supervised Nets. AISTA TS. Vol. 2, no. 3, (2015)
56. Srivastava, R.K., Greff, K., Schmidhuber, J.: Training very deep networks. In: NIPS’15: Proceedings of the 28th International Conference on Neural Information Processing Systems. Vol. 2, pp. 2377–2385 (2015)
57. Springenberg, J.T., Riedmiller, M.: Improving deep neural networks with probabilistic maxout units. In: International Conference on Learning Representations (ICLR), (2014)
58. Romero, A. et al.: Fitnets: Hints for thin deep nets. arXiv:1412.65 50 (2014)
59. Srivastava, N., Salakhutdinov, R.R.: Discriminative transfer learning with tree-based priors. Advances in Neural Information Processing Systems, (2013)
60. Ioffe, S., Christian, S.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:1502.031 67 (2015)
61. Chollet, F.K. https://github.com/fchollet/keras, (2016)
62. Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I.J., Bergeron, A., Bouchard, N., Bengio, Y.: Theano: new features and speed improvements. NIPS Workshop on Deep Learning and Unsupervised Feature Learning, (2012)
63. Huang, G. et al.: Densely connected convolutional networks. arXiv: 1608.06993 (2016)
64. https://www.kaggle.com/c/tiny-imagenet

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Md Zahangir Alom is a Bioinformatics Research Scientist at St. Jude Children’s Research Hospital, Memphis, TN, USA. He received his BS and MS degrees in Computer Science and Engineering from the University of Rajshahi, Bangladesh, and Chonbuk National University, South Korea, in 2008 and 2012, respectively, and he achieved his PhD degree in Electrical and Computer Engineering from University of Dayton in 2018. His research interests include machine learning, deep learning, medical imaging, computational pathology, and bioinformatics. He is a member of IEEE, member of INNS, and Digital Pathology Association (DPA) in USA.

Mahmudul Hasan obtained his bachelors and masters degree in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET). He received his PhD from Department of Computer Science and Engineering at the University of California, Riverside, in 2016. Currently, he has been working as a Research Scientist in Comcast Labs, Washington DC, USA. His research interests are computer vision: human activity recognition, object recognition, video description, and tracking; machine learning: active learning, deep learning, and reinforcement learning.

Chris Yakopcic is on the research faculty at the University of Dayton. He received his BS, MS, and PhD degrees in Electrical Engineering from the University of Dayton in 2009, 2011, and 2014, respectively. His research includes memristor modelling and circuit design and implementing neural and AI algorithms on low-power hardware. In 2013, he received the IEEE/INNS International Joint Conference on Neural Networks best paper award for a paper on memristor device modelling.

Tarek M. Taha received the BS degree from DePauw University, Greencastle, Indiana, in 1996 and the BSEE, MSEE, and PhD degrees in electrical engineering from the Georgia Institute of Technology, Atlanta, in 1996, 1998, and 2002, respectively. He is an Associate Professor of Electrical and Computer Engineering at the University of Dayton. His research interests include cognitive computing architectures, high-performance computing, and architectural performance modeling. He received the NSF CAREER Award in 2007.
Dr. Vijayan Asari is a Professor in Electrical and Computer Engineering and Endowed Chair in Wide Area Surveillance at the University of Dayton. He is also the director of the UD Vision Lab. Dr. Asari received his MTech and PhD degrees in electrical engineering from the Indian Institute of Technology, Madras. Dr. Asari received several teachings, research, advising and technical leadership awards.