Neural Computing and Applications (2020) 32:279–293
https://doi.org/10.1007/s00521-018-3627-6

ORIGINAL ARTICLE

Improved inception-residual convolutional neural network for object recognition

Md Zahangir Alom[1] · Mahmudul Hasan[2] · Chris Yakopcic[1] · Tarek M. Taha[1] · Vijayan K. Asari[1]

Received: 26 December 2017 / Accepted: 11 July 2018 / Published online: 4 August 2018

Abstract
Machine learning and computer vision have driven many of the greatest advances in the modeling of Deep Convolutional Neural Networks (DCNNs). Nowadays, most of the research has been focused on improving recognition accuracy with better DCNN models and learning approaches. The recurrent convolutional approach is not applied very much, other than in a few DCNN architectures. On the other hand, Inception-v4 and Residual networks have promptly become popular among computer the vision community. In this paper, we introduce a new DCNN model called the Inception Recurrent Residual Convolutional Neural Network (IRRCNN), which utilizes the power of the Recurrent Convolutional Neural Network (RCNN), the Inception network, and the Residual network. This approach improves the recognition accuracy of the Inception-residual network with same number of network parameters. In addition, this proposed architecture generalizes the Inception network, the RCNN, and the Residual network with significantly improved training accuracy. We have empirically evaluated the performance of the IRRCNN model on different benchmarks including CIFAR-10, CIFAR-100, TinyImageNet-200, and CU3D-100. The experimental results show higher recognition accuracy against most of the popular DCNN models including the RCNN. We have also investigated the performance of the IRRCNN approach against the Equivalent Inception Network (EIN) and the Equivalent Inception Residual Network (EIRN) counterpart on the CIFAR-100 dataset. We report around 4.53, 4.49 and 3.56% improvement in classification accuracy compared with the RCNN, EIN, and EIRN on the CIFAR-100 dataset respectively. Furthermore, the experiment has been conducted on the TinyImageNet-200 and CU3D-100 datasets where the IRRCNN provides better testing accuracy compared to the Inception Recurrent CNN, the EIN, the EIRN, Inception-v3, and Wide Residual Networks.

Keywords DCNN · RCNN · Inception network · Residual network · Deep learning

1 Introduction

Recently, deep learning using Convolutional Neural Networks (CNNs) has shown great success in the field of machine learning and computer vision. The CNNs provide state-of-the-art accuracy in various image recognition tasks including object recognition [1], segmentation [2], human activity analysis [3], image super resolution [4], object detection [5], tracking [6], image captioning [7], and scene understanding [5, 8]. Additionally, this approach has been applied passively in video processing tasks including video classification [9], video representation, and classification of human activity [10]. Deep learning is applied in sentiment analysis which is used for online movie recommendation systems, in addition to other applications [11]. Deep learning approaches are used in the field of machine translation

✉ Tarek M. Taha
ttaha1@udayton.edu

Md Zahangir Alom
alomm1@udayton.edu

Mahmudul Hasan
mahmud.ucr@gmail.com

Chris Yakopcic
cyakopcic1@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

and natural language understanding, and they achieve state-of-the accuracy in this application domain [12, 13]. Furthermore, this technique has been used extensively in the field of speech recognition [14]. Moreover, the deep learning technique is not limited to signal, natural language, image, and video processing tasks; it has been successfully applied in the field of game development [15, 16]. Machine intelligence provides improved performance in many different fields including calculation, chess, memory, and pattern matching, where as human intelligence still shows better performance in the fields of object recognition and scene understanding tasks. In the recent years, deep learning techniques (DCNNs in particular) have been providing outstanding performance for most of the tasks in computer vision. The DCNN is a hierarchical feature learning approach with multi-level and multi-scale abstraction of features, which aids in the learning of global contextual information from the input samples. However, there is still a gap that must be closed before human level intelligence can be achieved when performing visual recognition tasks. To reach human level performance during recognition tasks, a lot of research is dedicated to understanding the actual process of recognition, as well as understanding the tasks of the visual cortex in the human brain. Studies show that the human brain processes visual information using operations that are similar to convolution or filtering, activation, pooling, and normalization with recurrent connectivity in the visual cortex [17]. The recurrent connectivity of synapses in the human brain plays a big role for context modeling in visual recognition tasks [17, 18] (Fig. 1).

If we observe the structure of recently developed DCNN models, most functionalities are included to design better architectures which are successfully applied to segmentation, detection, and recognition tasks. However, the concept of Recurrent Convolution Layers (RCLs) is included in very few DCNN models, the most prominent being the Recurrent Convolutional Neural Network (RCNN) [19], a CNN with LSTM for object classification [20], a general Recurrent Multilayer Perceptron (RMLP)for dynamic control system [21] and the Inception RCNN (IRCNN) for object recognition [22]. On the other hand, Inception [23], and Residual [24, 25] architectures are commonly used for solving computer vision tasks. The common practice in the most recently developed Inception and Residual networks is to implement larger and deeper networks to archive better performance. As the model becomes larger and deeper, the parameters of the network are increased dramatically. As a result, the model becomes more complex to train and thus, more computationally expensive. Therefore, it is very important to design an architecture which provides better performance using reasonably fewer numbers of network parameters. While others are trying to implement bigger and deeper DCNN architectures like GoogLeNet [26], or a Residual Network with 1001 layers [24] to achieve high recognition accuracy on different benchmark datasets. We are presenting an improved version of the DCNN model inspired by the recently developed promising DCNN architectures like Inception-v4 [23], Residual [25], and the RCNN [19]. The proposed model not only ensures better recognition accuracy with same number of network parameters against other DCNN architectures, but also helps to improve the overall training accuracy. The contributions of this work are as follows:

- A new deep learning model named the Inception Recurrent Residual Convolutional Neural Network (IRRCNN) is proposed.
- Empirical evaluation of the performance of the proposed model against different DCNN models on different benchmark datasets such as CIFAR-10, CIFAR-100, TinyImageNet-200, and CU3D-100.
- Empirical investigation of the impact of the RCLs of the IRRCNN against that of the equivalent inception and inception-residual models on the CIFAR-100 and TinyImageNet-200 datasets.
- Large scale implementation to investigate the learning behavior of IRRCNN model and comparison against Inception-v3 and WRN models on the CU3D-100 dataset.

The rest of the paper has been organized as follows: Sect. 2 describes related work, and Sect. 3 presents the theoretical details of the IRRCNN model. Results and discussion are provided in Sect. 4, and conclusions and future work are discussed in Sect. 5.

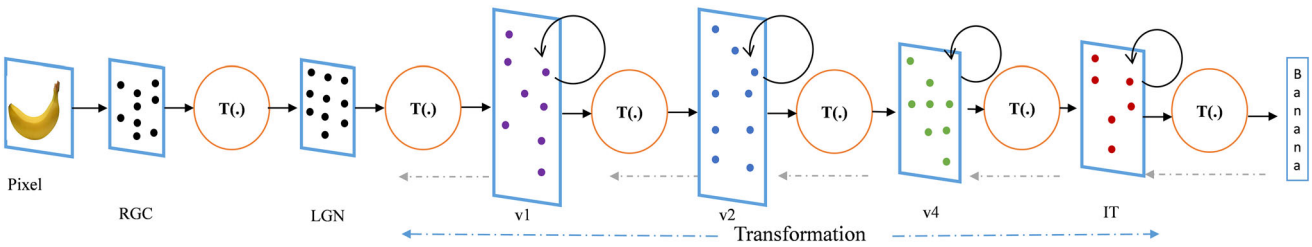

Fig. 1 Visual information processing pipeline of the human brain, where v1 though v4 represent the visual cortex areas. The visual context areas of v1 though v4 process information using recurrent techniques

2 Related work

Most of the breakthroughs in the field of computer vision (as well as the ImageNet challenges) have driven the development of the different DCNN architectures in recent years. The deep learning revolution began in 1998 with [27]. From then on, several different architectures have been proposed that have shown great success using many different benchmark datasets including MNIST, SVHN, CIFAR-10, CIFAR-100, ImageNet, and many more. Of the DCNN architectures, AlexNet [1], VGG [28], NiN [29], the All Convolutional Network [30], GoogLeNet [26], Inception-v4 [23], and the Residual Network [25] can be considered the most popular deep learning architectures due to their outstanding performance on different benchmarks for object classification tasks. In most cases, researchers experiment with different models such as NIN, the All Convolutional Network, VGG, GoogLeNet, Inception, and Residual networks, and then select the best model for their application based on the performance. Nevertheless, new models, hybrid models, and optimized versions of existing models have been proposed to achieve better accuracy with less network parameters in the last few years. The concept of Inception was introduced with GoogLeNet [26], and it won the most difficult ImageNet challenge for visual object recognition called the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2014 with remarkably few parameters. The main contribution of this network is to reduce the network parameters drastically when compared to the traditional CNN used in AlexNet. This model introduced a new technique called an inception layer. This approach is not only computationally convenient when compared to the traditional approach, but it also provided the best recognition accuracy in ILSVRC 2014. In terms of network parameters and memory, GoogLeNet needs only 4M whereas AlexNet needs around 60M [26]. An improved version of the Inception network was proposed by Szegedy et al. in 2015, where they scaled up the Inception model utilizing more computation with factorized convolution and aggressive regularization [23, 31]. This model shows a significant improvement in recognition accuracy on ILSVRC 2012. In 2015, He et al. proposed a new DCNN architecture called the Residual Network [25] and won the most difficult ILSVRC in 2015. This deep learning technique achieves state-of-the-art recognition accuracy on different benchmarks including ImageNet and CIFAR, as well as on object detection and segmentation tasks on PASCAL VOC and MSCOCO. This architecture is applied to different application domains including machine translation [32], speech synthesis [33], speech recognition [34] and audio classification [35]. Residual networks provide the possibility of building deep network architectures with thousands of layers resulting in significantly improved recognition accuracy [24]. However, improving just a fraction of a percentage in recognition accuracy requires almost doubling the number of layers in the networks. As a result, the number of model parameters and complexity increases. Therefore, training with very deep networks becomes very difficult due to diminishing feature reuse, which makes the networks very slow to train. Research has been conducted focusing on designing alternative models that produce the same level of recognition accuracy like SqueezeNet, which requires significantly less model parameters [36]. To overcome the problem of training complexity in residual networks, wide residual networks (WRN) have been proposed [37], where the width (number of feature maps) of the networks is increased instead of the depth (number of layers). In 2016, the aggregated residual network was also proposed which is a slight variant of the basic residual network structure [38].

Most existing research has been concentrated on improving recognition accuracy with different DCNN models. Out of the many modes, very few studies are using RCLs in their models. However, the recurrent approach is very important for context modeling in sequential images and videos. The RCNN structure was proposed for object recognition tasks by Ming et al. in 2015 [19]. This deep learning model contains several blocks of RCLs followed by a max-pooling layer. The global max-pooling layer is placed before the classification layer with Softmax at the end. This model provided state-of-the-art accuracy for object classification at that time [19]. In 2014, the Long-term Recurrent Convolutional Network (LRCN) was proposed for visual recognition and description by Donahue et al. [20]. This architecture contains of two popular techniques, the CNN and LSTM. The CNN technique is used for feature extraction, and LSTM is applied to observe how features vary with respect to time. This model shows outstanding performance for visual description [20]. Moreover, some research is being conducted that emphasizes on bridging gap between machine and human intelligence, where the proposed networks utilize recurrent concepts using residual network models [39]. Inception and Residual architectures are very prevalent in the computer vision community. The success of both architectures in the last few years has produced a new path of research that focuses on the discovery of even better models with better performance. Incorporating the new functionalities of RCLs into these state-of-the-art models improves overall recognition accuracy while utilizing the same number of the network parameters. This will have significant impact on the both computer vision and machine learning communities. In this paper, we have proposed an improved DCNN architecture based on Inception [23], Residual networks [25] and the RCNN architecture [19]. Therefore,

we call this model the Inception Recurrent Residual Convolutional Neural Network (IRRCNN).

3 IRRCNN architecture

The main objective of this model is to improve recognition performance using the same number or fewer computational parameters when compared to alternative equivalent deep learning approaches. In this model, the inception-residual units utilized are based on Inception-v4 [23]. The Inception-v4 network created by Szeged et al. in 2015 is a deep learning model that concatenates the outputs of the convolution operations with different sized convolution kernels in the inception block [23]. Inception-v4 is a simplified structure of Inception-v3 containing more inception modules using lower rank filters. Furthermore, Incpetion-v4 includes a residual concept in the inception network called the Inception-v4 Residual Network, which improves overall accuracy of recognition tasks. In the inception-residual network, the outputs of the inception units are added to the inputs of the respective units. The overall structure of the proposed IRRCNN model is shown in Fig. 2. From the figure, it can be clearly seen, that the overall model consists of several convolution layers, IRRCNN blocks, transition blocks, and a softmax at the output layer.

The most significant part of this proposed architecture is the IRRCNN block that includes RCLs, inception units, and residual units (shown in detail in Fig. 3). The inputs are fed into the input layer, then passed through inception units where RCLs are applied, and finally the outputs of the inception units are added to the inputs of the IRRCNN-block. The recurrent convolution operations perform with respect to the different sized kernels in the inception unit. Due to the recurrent structure within the convolution layer, the outputs at the present time step are added with the outputs of previous time step. The outputs at the present time step are then used as inputs for the next time step. The same operations are performed with respect to the time steps that are considered. For example, here $k = 2$ means that 3 RCLs are included in IRRCNN-block. In the IRRCNN-block, the input and output dimensions do not change, this is simply an accumulation of feature maps with respect to the time steps. As a result, the healthier features ensure that better recognition accuracy is achieved with the same number of network parameters.

The operations of the RCL are performed with respect to the discrete time steps that are expressed according to the RCNN [20]. Lets consider the x_l input sample in the l^{th} layer of the IRRCNN-block and a pixel located at (i,j) in an input sample on the k^th feature map in the RCL. Additionally, lets assume the output of the network $O^l_{ijk}(t)$ is at the time step t. The output can be expressed as follows:

$$O^l_{ijk}(t) = \left(w^f_k\right)^T * x^{f(i,j)}_l(t) + \left(w^r_k\right)^T * x^{r(i,j)}_l(t-1) + b_k \tag{1}$$

Here $x^{f(i,j)}_l(t)$ and $x^{r(i,j)}_l(t-1)$ are the inputs to the standard convolution layers and for the l^{th} RCL respectively. The w^f_k and w^r_k values are the weights of the standard convolutional layer and the RCL of the k^{th} feature map respectively, and b_k is the bias. The outputs of RCL are fed to the standard ReLU activation function f and are expressed as:

$$y = f\left(O^l_{ijk}(t)\right) = \max\left(0, O^l_{ijk}(t)\right) \tag{2}$$

Here f is the standard Rectified Linear Unit (ReLU) activation function. We have also explored the performance of this model with the Exponential Linear Unit (ELU) activation function in the following experiments. The outputs y of the inception units for the different size kernels and average pooling layer are defined as $y_{1\times1}(x)$, $y_{3\times3}(x)$, and $y^p_{1\times1}(x)$ respectively. The final outputs of Inception Recurrent Convolutional Neural Networks (IRCNN) unit are defined as $\mathcal{F}(x_l, w_l)$ which can be expressed as

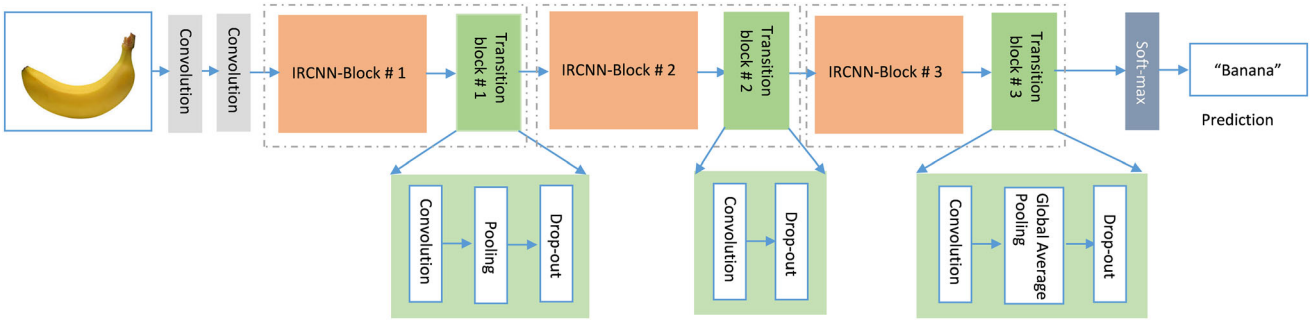

Fig. 2 The overall layer flow diagram of proposed IRRCNN consisting of the IRRCNN-Block, the IRRCNN-Transition block, and the Softmax layer at the end

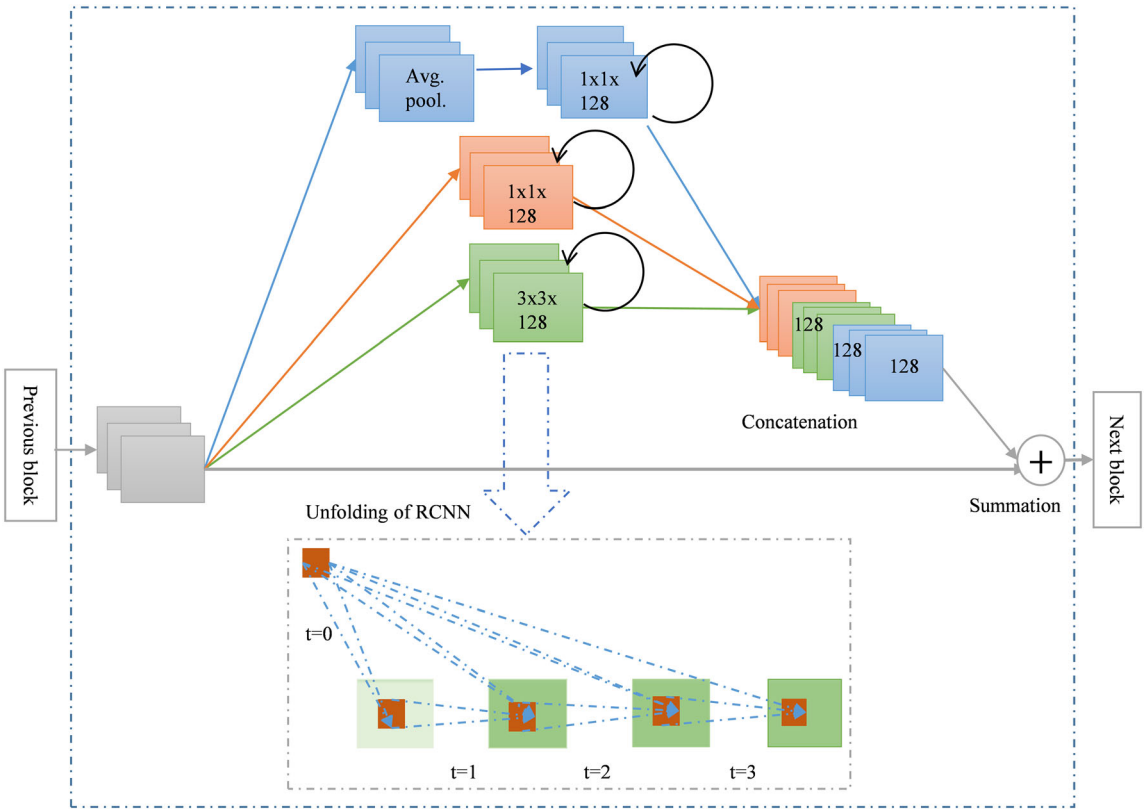

Fig. 3 The Inception Recurrent Residual Convolutional Neural Network (IRRCNN) block consisting of the inception unit at the top which contains recurrent convolutional layers that are merged by concatenation, and the residual units (summation of the input features with the outputs of the inception unit can be seen at the end of the block)

$$\mathcal{F}(x_l, w_l) = y_{1x1}(x) \odot y_{3x3}(x) \odot y^{p}_{1x1}(x) \tag{3}$$

Here $\odot$ represents the concatenation operation with respect to the channel or feature map axis. The outputs of the IRCNN-unit are then added with the inputs of the IRRCNN-block. The residual operation of the IRRCNN-block can be expressed by the following equation.

$$x_{l+1} = x_l + \mathcal{F}(x_l, w_l) \tag{4}$$

where x_{l+1} refers to the inputs for the immediate next transition block, x_l represents the input samples of the IRRCNN-block, w_1 represents the kernel weights of the lth IRRCNN-block, and $\mathcal{F}(x_l, w_l)$ represents the outputs from of lth layer of the IRCNN-unit. However, the number of feature maps and the dimensions of the feature maps for the residual units are the same as in the IRRCNN-block shown in Fig. 3. Batch normalization is applied to the outputs of the IRRCNN-block [40]. Eventually, the outputs of this IRRCNN-block are fed to the inputs of the immediate next transition block.

In the **transition block**, different operations are performed including convolution, pooling, and dropout, depending upon the placement of the transition block in the network. We did not include inception units in the transition block on the small-scale implementation for CIFAR-10 and CIFAR-100. However, we have applied inception units to the transition block during the experiment using the TinyImageNet-200 dataset and for the large-scale model which is the equivalent model of Inception-v3 [31]. The down-sampling operations are performed in the transition block where we perform max-pooling operations with a 33 patch and a 22 stride. The non-overlapping max-pooling operation has a negative impact on model regularization, therefore we used overlapped max-pooling for regularizing the network which is very important when training a deep network architecture [26, 41]. Late use of a pooling layer helps to increase the non-linearity of the features in the network, as this results in higher dimensional feature maps being passed through the convolution layers in the network. We have applied two special pooling layers in the model with three IRRCNN-blocks and a transition-block for the experiments that use the CIFAR-10 or CIFAR-100 dataset. We used only 1×1 and 3×3 convolution filters in this implementation, as inspired by the NiN [29] and Squeeze Net [34] models. This also helps to keep the number of network parameters at a minimum. The benefit of adding a 1×1 filter is that it helps to increase the non-linearity of the decision function

Table 1 Statistics for the datasets studied in these experiments

Dataset	Training samples	Val./testing samples	Total samples
CIFAR-10	50,000	10,000/10,000 (same)	60,000
CIFAR-100	50,000	10,000/10,000 (same)	60,000
TinyImageNet-200	100,000	10,000/10,000 (different)	120,000
CU3D-100	12,717	1413/4710 (different)	18,840

without having any impact on the convolution layer. Since the size of the input and output features does not change in the IRRCNN blocks, it is just a linear projection on the same dimension and non-linearity is added to the RELU and ELU activation functions. We used a 0.5 dropout after each convolution layer in the transition block. Finally, we used a softmax, or normalized exponential function layer at the end of the architecture. For input sample x, weight vector w, and k distinct linear functions, the softmax operation can be defined for the i^{th} class as follows:

$$p(y = i|x) = \frac{e^{x^T} w_i}{\sum_{k=1}^{K} e^{x^T} w_k} \tag{5}$$

This proposed IRRCNN model has been investigated through a set of experiments on different benchmark datasets and compared across different models.

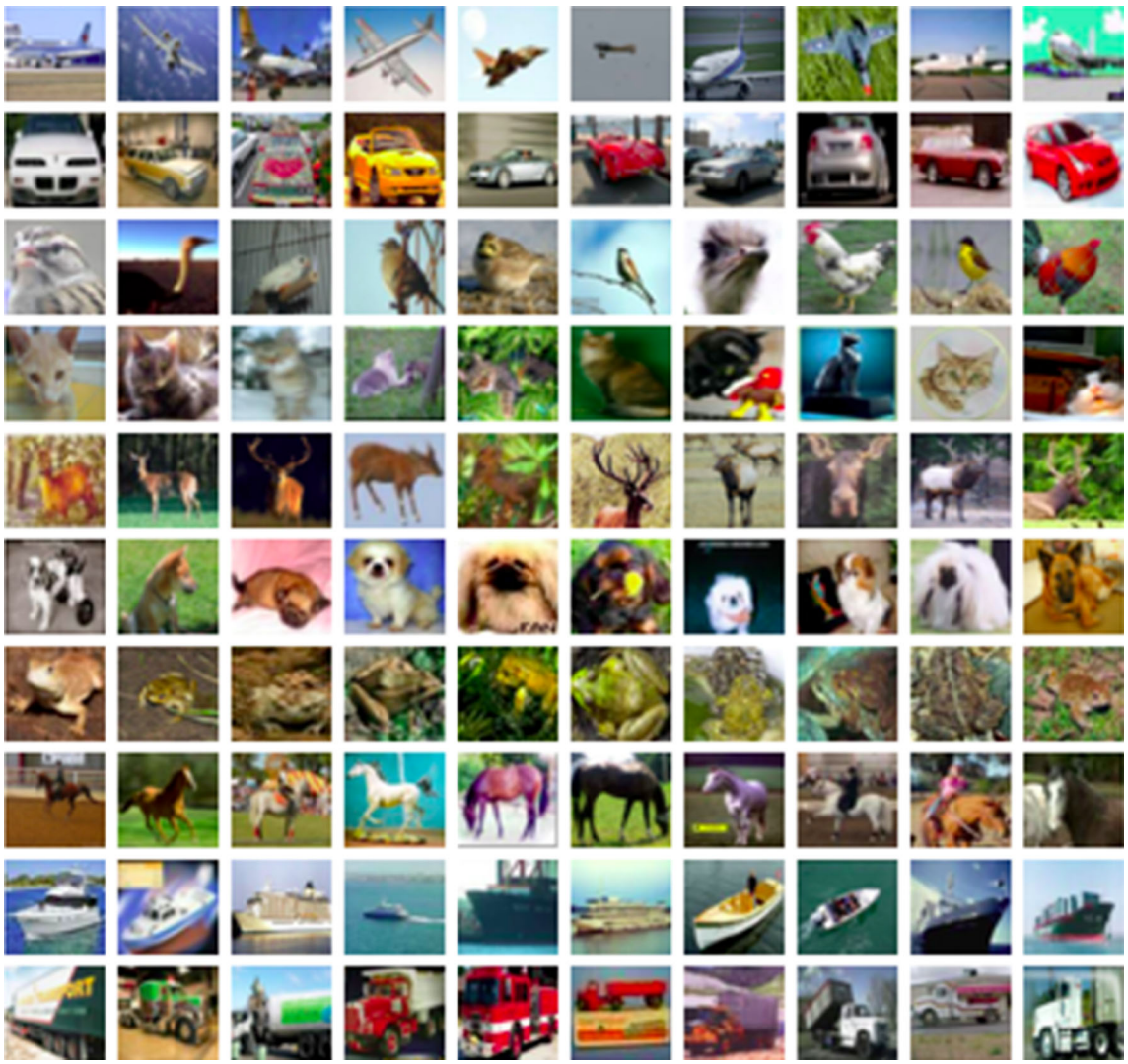

Fig. 4 Example images from the CIFAR-10 dataset

4 Experimental results and discussion

The proposed IRRCNN model has been evaluated using four different benchmark datasets: CIFAR-10 [36], CIFAR-100 [36], TinyImageNet-200 [38], and CU3D100 [35]. The dataset statistics are provided in Table 1. We used different validation and testing samples for the TinyImageNet-200 dataset. The entire experiment was conducted on a Linux environment with Keras [42] and Theano [43] at the backend running on a single GPU machine with an NVIDIA GTX-980Ti.

4.1 Experiments on CIFAR-10 and 100 datasets

In this experiment, we used two convolution layers at the beginning of the architecture, three IRRCNN blocks followed by three transition blocks, and one global average pooling and Softmax layer at the end. First, we evaluated the IRRCNN model using the stochastic gradient descent (SGD) technique with the Keras 2.0 [42] default initialization technique. We used momentum equal to 0.9 [39] and decay equal to $9.99 \times e^{-}7$ in this experiment. Second, we evaluated the same model with the Layer-sequential unit-variance (LSUV) initialization method [44] and the latest improved version of the optimization function called EVE [45]. The hyper parameters for the EVE optimization function are as follows: the value of learning rate (λ) is $1 \times e - 4$, decay (γ) is $1 \times e - 4$, $\beta_1 = 0.9$, $\beta_2 = 0.9$, $\beta_3 = 0.9$, $k = 0.1$, $K = 10$, and $\epsilon = 1 \times e - 08$. The values $(\beta_1, \beta_2) \in [0, 1)$ are exponential decay rates for moment estimation in Adam. The $\beta_3 \in [0, 1)$ is exponential decay rate for computing relative changes. The IRRCNN-block uses the $l_2 - norm$ for a weight regularization of 0.002. We used the ReLU activation function in the first experiment, and the ELU activation is used in the second experiment. In both experiments, we trained the networks for 350 epochs with a batch size of 128 for CIFAR-10 and 100.

CIFAR -10 The CIFAR-10 dataset is a benchmark dataset for object classification [36]. The dataset consists of 3232 color images split into 50,000 samples for training, and the remaining 10,000 samples are used for testing (classification into one of 10 classes) (Fig. 4). The experiment was conducted with and without data augmentation. When using data augmentation, we applied only random horizontal flipping. Using this proposed approach, we have achieved around 8.41% testing error without data augmentation and 7.37% testing error with augmented data (only horizontal flipping) using SDG techniques (Fig. 4).

The proposed model shows better recognition against most of the DCNN models displayed in Table 2. Furthermore, improved performance is observed in the IRCNN that used LSUV [44] initialization and the EVE [45] optimization function. The results show a testing error of around 8.17 and 7.11% without and with data augmentation respectively. It is also observed that the IRRCNN shows better performance when compared to the equivalent IRCNN model [21].

CIFAR -100 Another similar benchmark for object classification was developed in 2009 [36]. The dataset contains 50,000 samples for training and 10,000 samples for validation and testing. Each sample is a $32 \times 32 \times 3$ image, and the dataset has 100 classes. The proposed IRRCNN model was studied with and without data augmentation. During the experiment with augmented data, the SGD and LSUV [44] initialization approaches and the EVE optimization function were used [45]. In both cases, the

Table 2 Testing error (%) of the IRRCNN on CIFAR-10 object classification dataset without and with data augmentation

Methods	Number of Param.	C10	C10+
Maxout [46]	$> 6M$	11.6	9.38
NiN [29]	$> 1M$	10.4	8.81
DSN [47]	$> 1M$	9.69	7.97
CNN+Prob. maxout [48]	–	9.39	–
All-Conv. [30]	–	9.08	7.25
Highway Net. [49]	–	–	7.72
RCNN [19]	–	8.69	7.09
dasNet [50]	–	–	9.22
FitNet [51]	$> 2.5M$	–	8.39
Residual Net. [25]	–	–	7.51
IRCNN + SGD + ReLU	$\sim 3.5M$	8.41	7.37
IRCNN + LSUV + EVE + ReLU	$\sim 3.5M$	8.17	7.11
IRRCNN + SGD + ReLU	$\sim 3.5M$	8.14	7.11
IRRCNN + LSUV + EVE + ReLU	$\sim 3.5M$	**8.11**	**7.06**

For unbiased comparison, we have listed the accuracy stated in recent studies using a similar experimental setting. Here C10 refers without data augmentation and C10+ refers with data augmentation

Bold numbers indicate the lowest testing errors

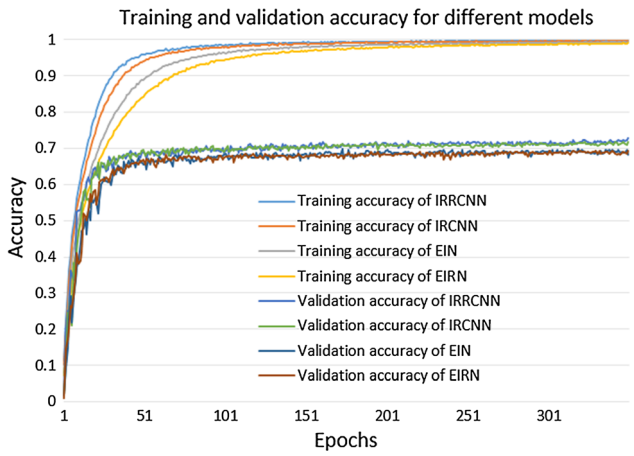

Fig. 5 Training and validation accuracy for IRRCNN, IRCNN, BIN, and BIRN on CIFAR-100. The vertical and horizontal axis represents accuracy and epochs respectively. Our proposed model shows the best recognition accuracy in all cases

proposed technique shows better recognition accuracy compared with different DCNN models including the IRCNN [21]. The validation accuracy of the IRRCNN model for both experiments on CIFAR-100 with data augmentation is shown in Fig. 5. The proposed IRRCNN model shows better performance in the both experiments when compared to the IRCNN [21], EIN, and EIRN models. The experimental results when using CIFAR-100 are shown in Table 3. The IRRCNN model provides better testing accuracy compared to many recently developed methods. We have achieved 72.78% recognition accuracy with LSUV+EVE which is around a 4.49% improvement compared to one of the baseline RCNN methods with almost the same number of parameters ($\sim 3.5M$) [19].

4.2 Impact of recurrent convolution layers

A question may arise here is there any advantage of the IRRCNN model against the EIRN and EIN architectures? The EIN and EIRN models are implemented with a similar architecture with same number of network parameters ($\sim 3.5M$). We used sequential convolution layers with the same time-step with the same size kernels instead of using RCLs for implementing the EIN and EIRN models. In addition, in the case of EIRN, we incorporated the residual concept with an Inception-block like Inception-v4 [23]. Furthermore, we have investigated the performance of the IRRCNN model against the RCNN with same number of parameters on the TinyImageNet-200 dataset.

A possible second question may arise Is the IRRCNN model providing better performance due to the use of advance deep learning techniques? It is noted that LSUV initialization approach applied to the DCNN architecture called FitNet4 achieved 70.04% classification accuracy on augmented data with mirroring and random shifts for CIFAR-100 [51]. In contrast, we only applied random horizontal flipping for data augmentation and achieved around 1.76% better recognition accuracy against FitNet4 [51].

The model accuracy for both training and validation are shown in Fig. 5. From the figure, it is clearly observed that this proposed model shows lower loss and highest recognition accuracy compared to EIN and EIRN, which proves the necessity of the proposed models. The testing accuracy of IRRCNN, IRCNN, EIN, and EIRN are shown in Fig. 6. It can be summarized that the proposed IRRCNN provides around 1.02, 4.49, and 3.56% improved testing accuracy compared to IRCNN, EIN, and EIRN respectively.

Table 3 Testing error (%) of the IRRCNN on the CIFAR-100 object classification dataset without and with data augmentation

Methods	Number of Param.	C100	C100+
Maxout [46]	$> 6M$	–	38.57
CNN+Prob. maxout [48]	$> 6M$	–	38.14
NiN [29]	$> 1.98M$	–	35.68
DSN [47]	–	–	34.57
dasNet [50]	–	–	33.78
All-Conv. [30]	–	–	33.71
RCNN-160 [19]	$1.87M$	–	31.75
Highway Net. [49]	–	–	32.24
FitNet [51]	–	–	35.04
BO with DNN [52]	–	–	27.40
IRCNN + SGD + ReLU	$\sim 3.5M$	34.13	31.22
IRCNN + LSUV + EVE + ReLU	$\sim 3.5M$	30.87	28.24
IRRCNN + SGD + ReLU	$\sim 3.5M$	33.07	29.21
IRRCNN + LSUV + EVE + ReLU	$\sim 3.5M$	**29.67**	**27.10**

For unbiased comparison, we have listed the accuracy provided by recent studies in a similar experimental setting. Here C100 refers without data augmentation and C100+ refers with data augmentation

Bold numbers indicate the lowest testing errors

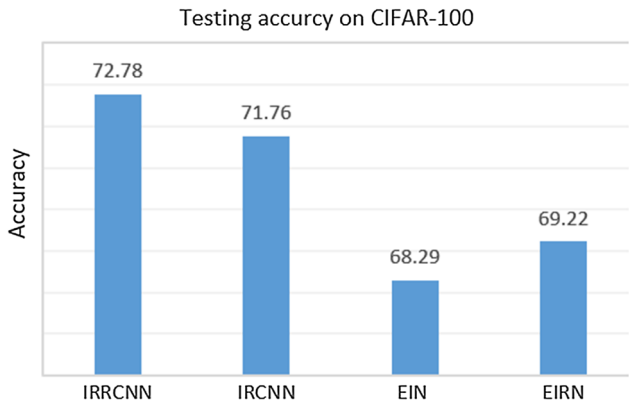

Fig. 6 Testing accuracy of the proposed IRRCNN model against IRCNN, EIN, and EIRN on the augmented CIFAR-100 dataset

4.3 Experiment on TinyImageNet-200

We also evaluated the proposed approach on the TinyImageNet-200 dataset [45]. This dataset contains 100,000 samples for training, 10,000 samples for validation, and 10,000 samples for testing. These images are sourced from 200 different classes of objects. Some of the example images are shown in Fig. 7. The main difference between the main ImageNet dataset and Tiny ImageNet is the images are down sampled from 224×224 to 64×64. There are some negative impacts of down-sampling, like loss of detail. Therefore, down sampling the images leads to ambiguity, which makes this problem even harder and this effects overall model accuracy. For this experiment, we used the IRRCNN model with two general convolution layers with a 3×3 kernel at the beginning of the network followed by sub-sampling layer with 3×3 convolution using a stride of 2×2. After that, four IRRCNN blocks are used followed by four transition blocks. Finally, a global average pooling layer is used followed by a softmax layer.

We have experimented with the IRRCNN, IRCNN, equivalent RCNN, EIN, and EIRN using the TinyImageNet-200 dataset. The training accuracy of this experiment is shown in Fig. 8. The proposed IRRCNN model provides better recognition accuracy during training compared to equivalent models including IRCNN, EIN, and EIRN with almost the same number of network

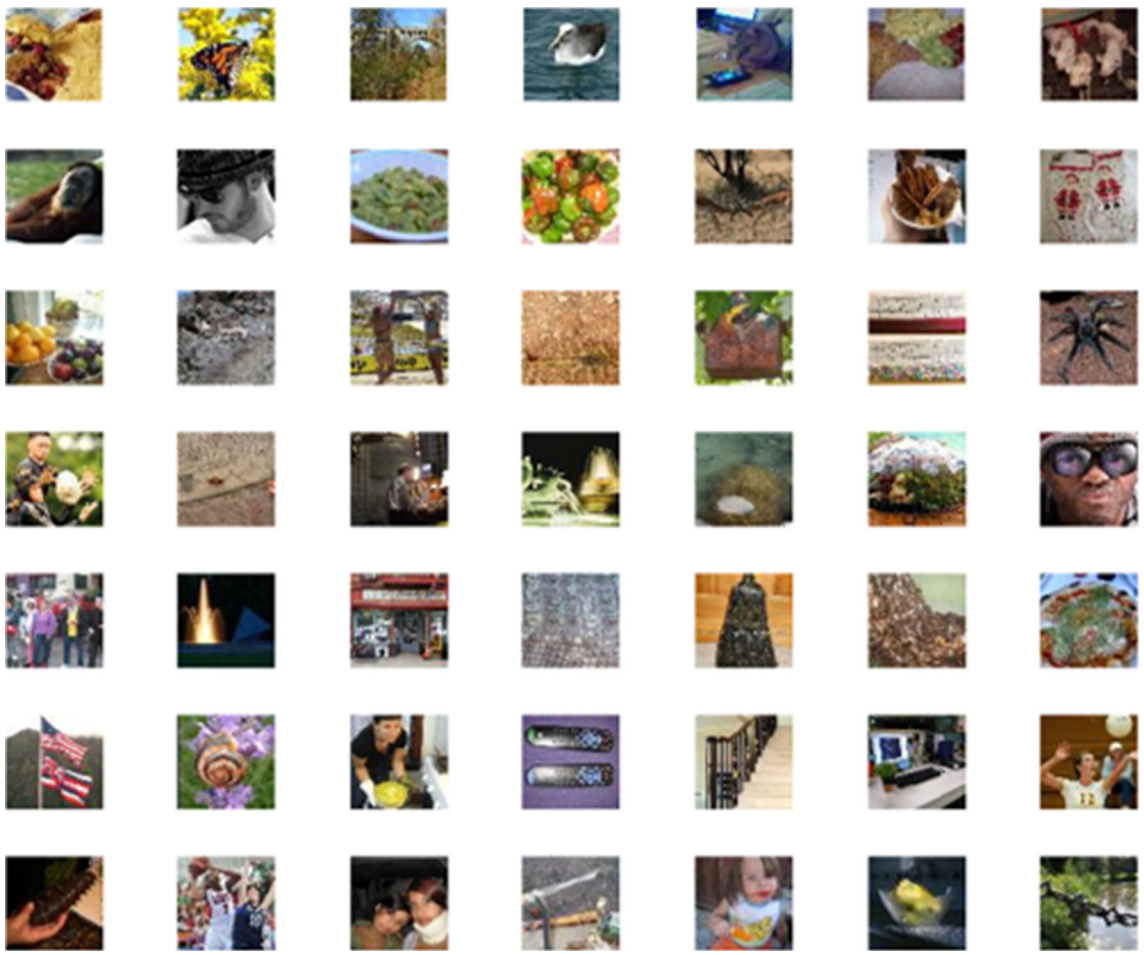

Fig. 7 Sample images from the TinyImageNet-200 dataset

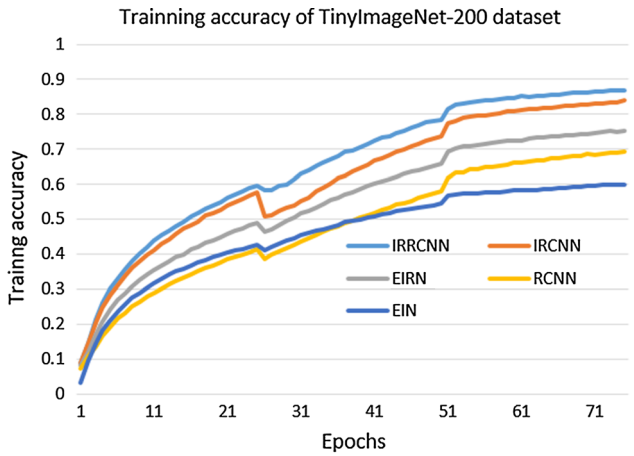

Fig. 8 Training accuracy during training for TinyImageNet-200 dataset

parameters ($\sim 15M$). Generally, DCNN takes a lot of time and power when training a reasonably large model. The inception-residual networks with RCLs significantly reduce training time with faster convergence and better recognition accuracy. The validation accuracy for all of these models is shown in Fig. 9. We have evaluated our proposed approach for both Top-1% and Top-5% testing accuracy as shown in Fig. 10. From the bar graph, the impact of recurrent connectivity is clearly observed, and we have achieved 52.23% top-1% testing accuracy whereas the EIRN and EIN show 51.14 and 45.63% top-1% testing accuracy. The same behavior is observed for Top-5% accuracy as well. The IRRCNN provides better testing accuracy when compared against all other models in both cases which absolutely displays the robustness of the proposed deep learning architecture.

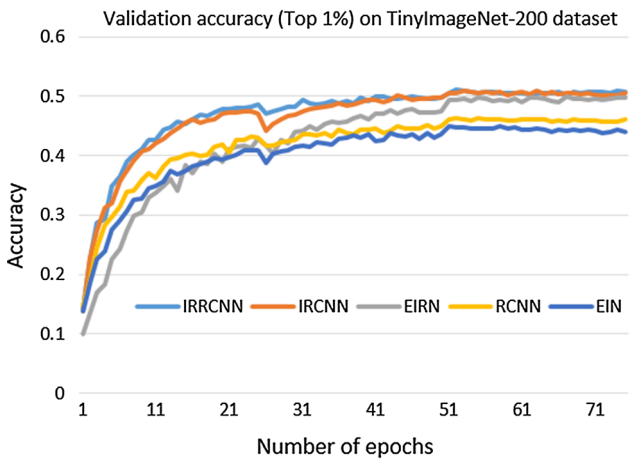

Fig. 9 Validation accuracy on the TinyImageNet-200 dataset

4.4 IRRCNN versus Inception-v3 and WRN models

We have evaluated the IRRCNN model with large scale implementation against the Ineption-v3 and WRN networks. The IRRCNN model is implemented with a similar structure to the Incpeiton-v3 for impartial comparison. We have used the default implementation of Keras version 2.0 and just incorporated the RCLs for $k = 2$, where $(k = 2)$, which means one forward convolution and two RCLs are used in the Inception units and a residual layer is used at the end of the blocks. The WRN is implemented according to the structure described in [32]. We trained the networks with the SGD optimization method with momentum 0.9 for 25 epochs.

4.4.1 CU3D-100 dataset

Another high quality visual object recognition dataset with well-controlled images (e.g., object invariance, feature complexity) is CU3D-100, which is suitable for evaluation of new deep learning algorithms. This dataset contains 18,840 color images in total that have a dimension of $64 \times 64 \times 3$ with 20 samples per exemplar [35]. Figure 11 shows some example images from the CU3D-100 dataset.

The images in this dataset are three-dimensional views of real-world objects normalized for different positions, orientations, and scales. The rendered images have a 400 depth rotation about the y-axis (plus a horizontal flip), a 200 tilt rotation about x-axis, and an 800 overhead lighting rotation. We used 75% percent of the images for training and the remaining 25% of the images for testing, which were selected randomly from the whole dataset. The example images in the fish category with different lighting conditions and affine transformations are shown in Fig. 12.

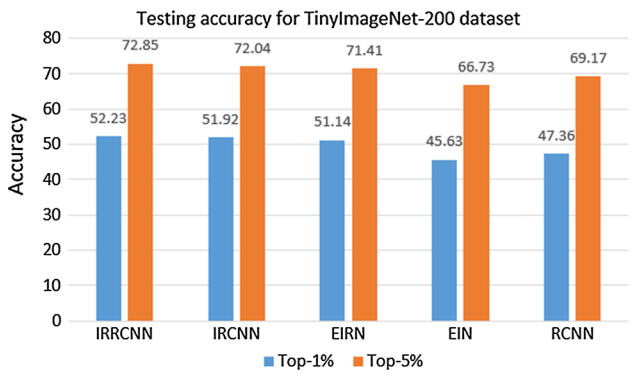

Fig. 10 Top-1% and Top-5% testing accuracy on TinyImageNet-200 dataset

Fig. 11 Example images of the CU3D-100 dataset

4.4.2 Experimental results on CU3D-100 dataset

We have conducted two different experiments, and in first experiment, the models are trained using the scratch and transfer learning approach. The pretrained weights from the ImageNet dataset are used for the second experiment. Both experiments are conducted using the CU3D-100 dataset. We considered an implementation of IRRCNN, which is an equivalent of Inception-v3 that contains $\sim 19.74M$ and $\sim 21.25M$ network parameters respectively. The WRN model consists of deep and wide factors $n = 6$ and $k = 6$ respectively, and contains $\sim 31.25M$ network parameters [32, 53]. The entire dataset was first divided into two sets where 75% of the (14,130) samples are used for training and validation, and the remaining 25% (4710) of the samples are used for testing. For the first experiment, using 14,130 samples, 10% of the samples are used for validation during training. The training and validation accuracies for 25 epochs are shown in Fig. 13. Figure 13 shows that the IRRCNN model exhibits lower error during training and validation when compared to the Incpeiton-v3 and WRN models.

In the testing phase of the first experiment, we have achieved 99.81, 99.13, and 98.51% testing accuracy with IRRCNN, Inception-v3, and WRN respectively. The IRRCNN model shows 0.68 and 1.30% higher testing accuracy against Inception-v3 and WRN respectively. A recently published paper with sparse parameterization back-propagation in a network with recurrent layers reported about 94.6% testing accuracy on the CU3D-100 dataset [35], which is around 5.24% less testing accuracy compared to this proposed IRRCNN model. In the second experiment, the pre-trained ImageNet weights are used as initial weights for IRRCNN and Inception-v3 models

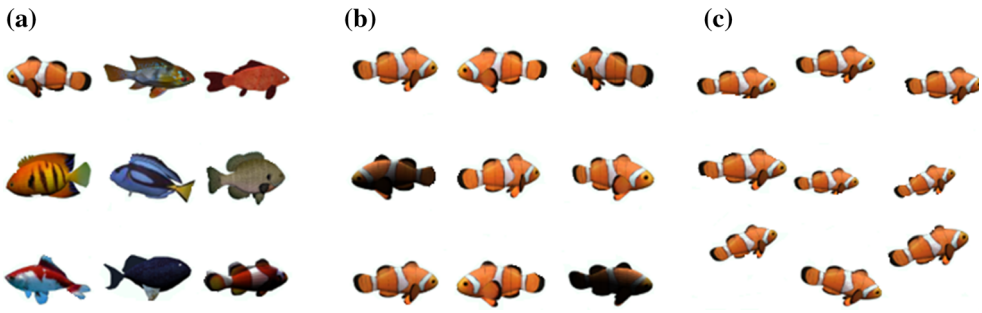

Fig. 12 Sample images displaying **a** nine examples from the fish category, **b** nine depth, tilt, and lighting variations of the fish category, and **c** nine affine transformation images for a single view

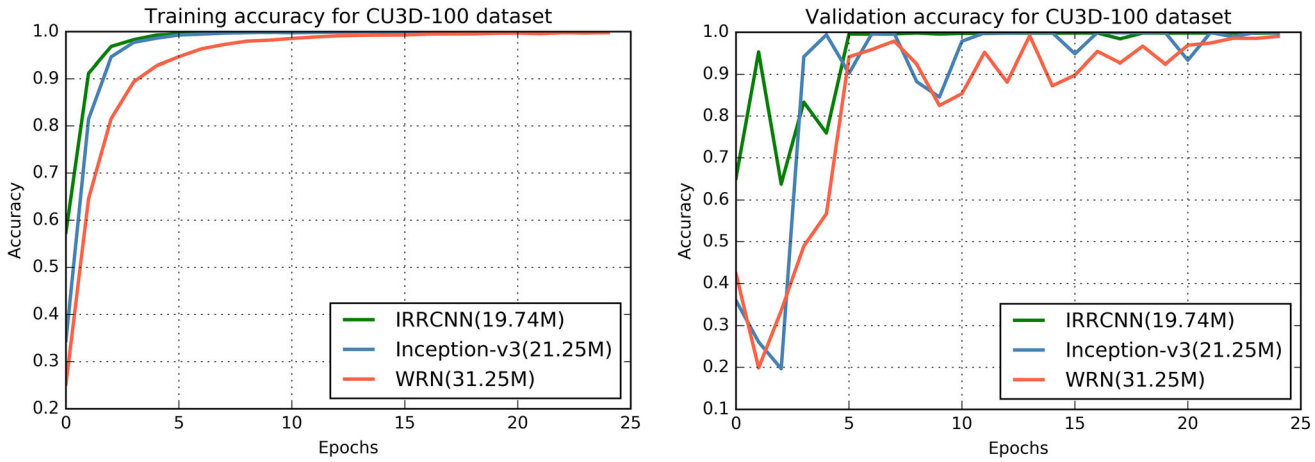

Fig. 13 Training and validation error with respect to the number of epochs on the CU3D-100 dataset: left figure shows training accuracy and right figure represents validation accuracy

where we have trained only a few of the layers at the top of the models. The pretrained weights were taken from GitHub [54]. The proposed model gives about 98.84% testing accuracy whereas Inception-v3 model gives 92.16% testing accuracy on the CU3D-100 dataset. The IRRCNN model shows around 6.68% better testing accuracy compared to the similar Inception-v3 model and clearly demonstrates the impact of RCL and residual layers in the models. This experiment also proves the robustness of the IRRCNN model when dealing with scale invariance, position and rotation invariance, and different lighting condition input samples.

4.4.3 Trade-off between split ratio versus training, validation,and testing errors

To further investigate the performance of the proposed IRRCNN model, the trade-o between the split ratio versus performance is investigated against Inception-v3 [31] and WRN [32]. During this experiment, we used different split ratios including [0.9, 0.7, 0.5, 0.3, and 0.1]. The number of training and validation samples are taken according to the split ratio where the number of training samples is increased and the number of validation samples is decreased in the trials respectively. For example, a split ratio of 0.9 refers to only 10% of the samples (1423) being used for training and remaining 90% of the samples (12815) are used for validation, while a split ratio 0.7 means 30% of the samples are used for training and the remaining 70% of the samples are used for validation and so on. However, it can be also observed from the Fig. 13 that the models converged after 22 epochs. Therefore, in each trial, we considered 25 epochs and the error here is the average training and validation error for the last five epochs.

Figure 14 shows the training and validation errors with respect to split ratios. This figure shows that the proposed IRRCNN model shows less training and validation errors for five different trials in both cases. These results clearly demonstrate that the IRRCNN is more capable at extracting, representing, and learning features during the training phase which ultimately helps to ensure better testing performance. In each trial, we tested the models with the remaining 25% of the samples, and the testing errors are shown in Fig. 15. From this figure, it is clear that IRRCNN shows the lowest error for almost all trails compared to Inception-v3 [31] and WRN [32, 53].

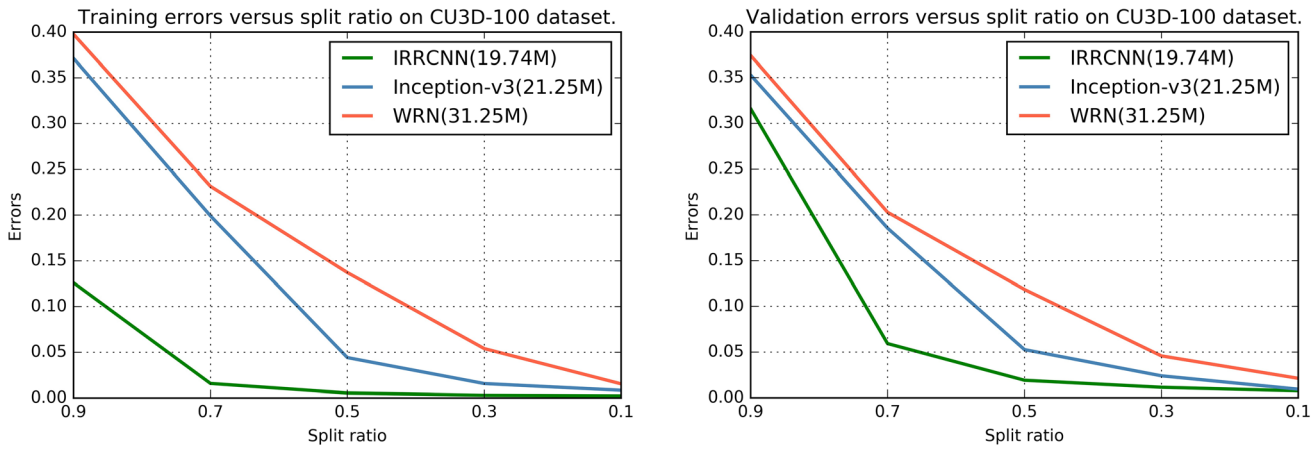

Fig. 14 The training and validation errors versus split ratio for five different trials on CU3D-100 dataset

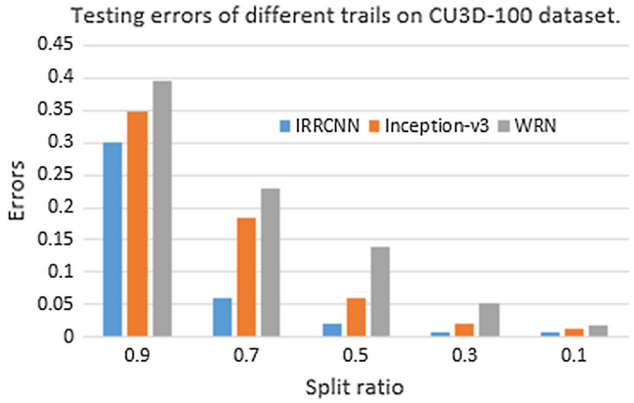

Fig. 15 Testing accuracy of the IRRCNN model compared to that of the Inception-v3 and WRN networks for five trials on the CU3D-100 dataset

4.5 Computational time

The computational time of the proposed IRRCNN and other equivalent models for different datasets are shown in Table 4.

4.6 Introspection

From our investigation, we have observed that the proposed IRRCNN model converges faster when compared to the RCNN, EIR, EIRN, and IRCNN models which are clearly evaluated using a set of experiments. The proposed techniques provide promising recognition accuracy during the testing phase with the same number of network parameters compared with other models. In this implementation, we have augmented input samples by applying only random horizontal flipping. From our observation, the proposed model will provide even better recognition accuracy with more augmentations including transition, central crop, and ZCA.

5 Conclusion

In this paper, we have proposed the Inception Recurrent Residual Convolutional Neural Network (in short IRRCNN) for object recognition where we have utilized the power of recurrent convolution neural layers for context modulation based on the Inception and Residual Network architectures. The experimental results show promising recognition accuracy compared with different Deep Convolutional Neural Network (DCNN) models on different benchmarks including CIFAR-10, CIFAR-100, TinyImageNet-200, and CU3D-100. However, the proposed model has been evaluated with different advanced training approaches including SGD, initialization with Layer-sequential unit-variance (LSUV), and the recently proposed optimization methods of EVE. The IRRCNN model with LSUV and EVE achieved a promising object recognition accuracy of 72.78% on the CIFAR-100 dataset which is about a 4.53% improvement when compared to the Recurrent Convolutional Neural Network (RCNN)

Table 4 Computational training time per epoch for four datasets

Model	Dataset	Time/epoch (in s)
IRRCNN/IRCNN/EIN/EIRN	CIFAR-10	~ 422
IRRCNN/IRCNN/EIN/EIRN	CIFAR-100	~ 425
IRRCNN/IRCNN/EIN/EIRN	TinyImageNet-200	~ 765
IRRCNN/Inception-V3 /WRN	CU3D-100	~ 978

[20]. In addition, our model provides about 4.49 and 3.56% improvement in recognition accuracy when compared with Equivalent Inception Networks (EIN) and Equivalent Inception-Residual Networks (EIRN) on the CIFAR-100 dataset. We have achieved better recognition accuracy with IRRCNN when compared to EIRN, EIN, RCNN, and IRCNN on the TinyImageNet-200 dataset. Furthermore, the trade-off between split ratio and training and validation errors is calculated to investigate the learning behavior of the IRRCNN, Inception-v3, and WRN models. The proposed IRRCNN shows better learning capability with less error during the training and testing phases for all trails. The IRRCNN shows 0.68 and 1.30% better testing accuracy compared to Inception-v3 and WRN. Based on all experimental evaluations, it is clearly observed that the proposed architecture shows better performance and accelerates the training process, which is a big issue right now for training large scale deep learning systems. In the future, we would like to improve this model and explore segmentation and detection tasks.

References

1. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems
2. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3431–3440
3. Toshev A, Szegedy C (2014) Deeppose: human pose estimation via deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1653–1660
4. Dong C, Loy CC, He K, Tang X (2014) Learning a deep convolutional network for image super-resolution. In: European conference on computer vision. Springer, Berlin, pp 184–199
5. Zhou B, Lapedriza A, Xiao J, Torralba A, Oliva A (2014) Learning deep features for scene recognition using places database. In: Advances in neural information processing systems, pp 487–495
6. Wang N et al (2015) Transferring rich feature hierarchies for robust visual tracking. To further investigate the performance of the proposed IRRCNN model. arXiv preprint arXiv:1501.04587
7. Mao J et al (2014) Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv preprint arXiv:1412.6632
8. Shankar S, Garg VK, Cipolla R (2015) Deep-carving: discovering visual attributes by carving deep neural nets. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3403–3412
9. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1725–1732
10. Ballas N et al (2015) Delving deeper into convolutional networks for learning video representations. arXiv preprint arXiv:1511.06432
11. RojasBarahona Lina Maria (2016) Deep learning for sentiment analysis. Lang Linguist Compass 10(12):701–719
12. Collobert R, Weston J (2008) A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning. ACM
13. Manning CD et al (2014) The Stanford CoreNLP natural language processing toolkit. ACL (System Demonstrations)
14. Geoffrey Hinton et al (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29(6):82–97
15. Mnih V et al (2013) Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602
16. Lillicrap TP et al (2015) Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971
17. DiCarlo JJ, Zoccolan D, Rust NC (2012) How does the brain solve visual object recognition? Neuron 73(3):415–434
18. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bull Math Biophys 5(4):115–133
19. Liang M, Hu X (2015) Recurrent convolutional neural network for object recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition
20. Donahue J, Hendricks LA, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: CVPR
21. Fernandez Benito, Parlos AG, Tsai WK (1990) Nonlinear dynamic system identification using artificial neural networks (ANNs). In: 1990 IJCNN international joint conference on neural networks. IEEE
22. Alom MZ, Hasan M, Yakopcic C, Taha TM (2017) Inception recurrent convolutional neural network for object recognition. arXiv:1704.07709
23. Szegedy C et al (2016) Inception-v4, Inception-Resnet and the impact of residual connections on learning. arXiv preprint arXiv:1602.07261
24. He K et al (2016) Identity mappings in deep residual networks. In: European conference on computer vision. Springer, Berlin
25. He K et al (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition
26. Szegedy C et al (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition
27. LeCun Y et al (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
28. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
29. Lin M, Chen Q, Yan S (2013) Network in network. arXiv preprint arXiv:1312.4400
30. Springenberg JT et al (2014) Striving for simplicity: the all convolutional net. arXiv preprint arXiv:1412.6806
31. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2818–2826
32. Zagoruyko S, Komodakis N (2016) Wide residual networks. arXiv preprint arXiv:1605.07146
33. Xie S, Girshick R, Dollr P, Tu Z, He K (2016) Aggregated residual transformations for deep neural networks. arXiv preprint arXiv:1611.05431
34. Iandola FN et al (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and $<$ 0.5 MB model size. arXiv preprint arXiv:1602.07360
35. Liao Q, Poggio T (2016) Bridging the gaps between residual learning, recurrent neural networks and visual cortex. arXiv preprint arXiv:1604.03640

36. O'Reilly RC, Wyatte D, Herd S, Mingus B, Jilk D (2013) Recurrent processing during object recognition. Front Psychol 4(124):1–14
37. Krizhevsky A (2009) Learning multiple layers of features from tiny images. Technical report
38. Tiny ImageNet (2017) https://tiny-imagenet.herokuapp.com/. Accessed Dec 2017
39. Ilya Sutskever et al (2013) On the importance of initialization and momentum in deep learning. ICML 3(28):1139–1147
40. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167
41. Wan L et al (2013) Regularization of neural networks using dropconnect. In: Proceedings of the 30th international conference on machine learning (ICML-13)
42. Keras CF (2016) https://github.com/fchollet/keras. Accessed Jan 2017
43. Bastien F, Lamblin P, Pascanu R, Bergstra J, Goodfellow IJ, Bergeron A, Bouchard N, Bengio Y (2012) Theano: new features and speed improvements. NIPS workshop on deep learning and unsupervised feature learning
44. Mishkin D, Matas J (2015) All you need is a good init. arXiv preprint arXiv:1511.06422
45. Koushik J, Hayashi H (2016) Improving stochastic gradient descent with feedback. arXiv preprint arXiv:1611.01505
46. Goodfellow IJ et al (2013) Maxout networks. ICML 3(28):1319–1327
47. Lee C-Y et al (2015) Deeply-supervised nets. AISTATS 2(3):562–570
48. Springenberg JT, Riedmiller M (2014) Improving deep neural networks with probabilistic maxout units. In: International conference on learning representations (ICLR)
49. Srivastava RK, Greff K, Schmidhuber J (2015) Training very deep networks. In: Advances in neural information processing systems (Highway Network)
50. Stollenga MF et al (2014) Deep networks with internal selective attention through feedback connections. In: Advances in neural information processing systems
51. Romero A et al (2014) Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550
52. Snoek J, Rippel O, Swersky K, Kiros R, Satish N, Sundaram N, Adams RP (2015) Scalable Bayesian optimization using deep neural networks. In: ICML, pp 2171–2180
53. https://gist.github.com/kashif/0ba0270279a0f38280423754cea2ee1e. Accessed Dec 2017
54. https://github.com/fchollet/deep-learning-models/releases. Accessed July 2017