Information Sciences 482 (2019) 63–72

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

# Multipath-DenseNet: A Supervised ensemble architecture of densely connected convolutional networks

Bilal Lodhi, Jaewoo Kang*

*Department of Computer Science and Engineering, Korea University, Seoul, South Korea*

ARTICLE INFO

*Article history:*
Received 2 August 2018
Revised 20 November 2018
Accepted 5 January 2019
Available online 6 January 2019

*Keywords:*
Image classification
Neural network
Deep-learning

ABSTRACT

Deep networks with skip-connections such as ResNets have achieved great results in recent years. DenseNet exploits the ResNet skip-connections by connecting each layer in convolution neural network to all preceding layers and achieves state-of-the-art accuracy. It is well-known that deeper networks are more efficient and easier to train than shallow or wider networks. Despite the high performance of very deep networks, they are limited in terms of vanishing gradient, diminishing forward flow, and slower training time. In this paper, we propose to combine the benefits of the depth and width of networks. We train supervised independent shallow networks on the same input in a block fashion. We use a state-of-the-art DenseNet block to increase the number of paths for gradient flow. Our proposed architecture has several advantages over other deeper networks including DenseNet; our architecture which we call Multipath-DenseNet is deeper as well as wider, reduces training time, and uses a smaller number of parameters. We evaluate our proposed architecture on the following four object recognition datasets: CIFAR-10, CIFAR-100, SVHN, and ImageNet. The evaluation results show that Multipath-DenseNet achieves significant improvement in performance over DenseNet on the benchmark datasets.

## 1. Introduction

Convolutional networks have been employed in research for many years. They have been successfully applied to image processing [1], natural language processing (NLP) [2], and recommender systems [3]. Research on convolutional neural networks has resulted in outstanding algorithms such as LeNet [4], AlexNet [5], VGGNet [6], ResNet [7], and GoogLeNet [8].

Highway [9] and ResNet [7] are considered to be the pioneer networks that were proposed to extract features from over 100 layers. Training very deep neural networks can be challenging due to the vanishing gradient problem. ResNet proposed skip-connection for deeper layers in order to make very deep neural networks. Stochastic depth algorithm proposed by [10] proved that depth is not the only parameter behind the success of residual networks (ResNets). The authors [10] proposed to shorten network depth by randomly skipping the layers in residual networks. Wide residual network [11] followed the similar hypothesis that depth is not the only important parameter. They shorten the network depth too but increased the number of features at individual layer of the network that makes a wider neural network. Wide residual Network which has depth 50 times smaller than that of ResNet, outperformed the original ResNet. Furthermore, FractalNet [12] improved

* Corresponding author.
*E-mail addresses:* blodhi@korea.ac.kr (B. Lodhi), kangj@korea.ac.kr (J. Kang).

https://doi.org/10.1016/j.ins.2019.01.012
0020-0255/© 2019 Elsevier Inc. All rights reserved.

the accuracy and depth of the convolutional neural network without the traditional residual connections, suggesting that a filter and a non-linear function has to be applied on data before it is sent to the successive layers.

Interestingly, residual networks do not resolve the vanishing gradient problem [13]. Instead, the problem is avoided by introducing shorter paths to carry the gradient up to the last layer of the network. These shorter paths actually behave like ensembles of shallow networks and are not strongly dependent on each other. It was proven that removal of a layer does not effect the performance of ResNet. However performance of VGGNet, which has only one path to accommodate the gradient, drops to random chance [13].

DenseNet [14] exploits the residual connections in residual network (ResNet). Each layer in DenseNet receives input from all preceding layers and forwards its output to all subsequent layers in a block. DenseNet also concatenates feature maps to give variety in the inputs of the subsequent layers, unlike the summation in residual networks. This feature reuse (feature concatenation) operation creates many paths, resulting in an exponential increase in memory requirements and computational complexity.

Another observation about ultra-deep networks is that a significant number of deeper layers are required to slightly increase the accuracy of residual networks [15]. The requirement of deeper layers is computationally expensive in DenseNet as compared to ResNet. In DenseNet, an increase in depth of a denseblock or an addition of a new denseblock exponentially increases the memory requirements and computational complexity. This increment, in depth, either makes a small improvement in the network performance or worsens it; increasing the parameters of memory efficient architecture of DenseNet (DenseNet-BC) (increasing the growth rate from 12 to 24 and the depth from 100 to 250) achieves a 5.92% to 5.19% decrease in the error rate on the CIFAR-10 dataset, while the parameters size becomes 19 times bigger (0.8M to 15.3M). Similarly, increasing the growth rate of DenseNet-100 from 12 to 24 increases the parameters from 7.0M to 27.2M. However, the error rate slightly increases from 5.77% to 5.83% on the CIFAR-10 dataset.

It is also shown in [13] that most of the gradient is from 10 to 34 layers deep in a residual network of 110 layers. To address the aforementioned shortcomings of ResNet [7], wide residual network [11] and DenseNet [14], we developed an architecture called Multipath-DenseNet, which is wider and has a suitable depth and more shorter-medium paths for efficient gradient flow.

The main contribution of the paper is fourfold:

- We propose an ensemble network architecture to improve the performance of state-of-the-art DenseNet.
- A supervising transformation block is added at the end of Multipath-denseblocks to learn the intermediate features generated by the individual multipath-denseblock.
- Multipath-DenseNet's architecture proves that depth is not the only factor in achieving high performance. Additionally, we illustrate that extracting low-level features from deeper layers of DenseNet overfits the easy tasks in dataset while affecting the performance on complex tasks.
- Our proposed architecture is evaluated on four benchmark datasets: CIFAR-10, CIFAR-100, SVHN, and ImageNet (downsized to $32 \times 32$).

In our experiments, we found that making the network length longer than the threshold does not make much improvement in the performance of a given network, while the features learned from the good gradient layers yield better accuracy. Moreover shallow networks requires less training time (reduced complexity). Multipath-DenseNet improved the accuracy with fewer training parameters and less computational complexity. The architecture of Multipath-DenseNet is both wide and deep. Parallel dense blocks behave like a wide neural network and depth of each dense block represents the number of layers in a deep neural network.

## 2. Related work

The ImageNet challenge [16] resulted in many successful networks in the field of convolutional neural network. AlexNet, VGG, GoogLeNet, and ResNet, which range from deep to deeper, improved performance on classification tasks. Also, Very Deep networks extract high-level features while attaining high accuracy. However, these networks suffer from convergence complexity, overfitting, and vanishing gradient issues [17,18].

Highway Networks [9] based on Long short-term memory (LSTM) recurrent networks [19] were proposed to carry the gradient to the deeper layers of the network. Two gating units are used to keep the input unaltered in a network layer. ResNet [7] is considered as a simplified version of a Highway Network, and has achieved significant improvement in performance. The main idea of these networks (ResNet and Highway) is to keep the input closer to the output. Similarly, Google also adopted the idea and proposed to add auxiliary classifiers to intermediate layers of GoogLeNet, stating that the values on these layers are more reliable since they are extracted from the layers in which the gradient carries more information [8]. Moreover, the authors of stochastic depth algorithm [10] propose randomly skipping the layers entirely in ResNet to bring the input closer to the output.

The authors in [15] propose to divide a deeper ResNet into multiple same size independent ResNets. These independent networks are trained separately, and information is added by a residual connection. These networks are simpler but achieve an accuracy similar to that of ResNet [15]. DenseNet [14] exploits the residual connection of ResNet and connects all of the layers to each other. Although DenseNet obtains higher accuracy than many networks, increasing the number of feature maps at each layer of DenseNet adds an excessive complexity to the network. Moreover, [20] proposed a modularized network architecture of ResNet for multiple paths in the network called as ResNeXt. But there can be one one dominant path in their proposed architecture.

As mentioned earlier, DenseNet obtains state-of-the-art accuracy but suffers from exponential parameter growth. A bottleneck architecture was proposed to alleviate the problem. Although DenseNet is parameter efficient and obtains good accuracy, it has the following drawbacks: a) DenseNet requires quadratic memory with respect to their depth; b) DenseNet has a great amount of redundant information which makes network convergence difficult; c) many residual connections increase the chances of overfitting the data; d) information reuse in training prevents the network from learning high-level features for complex tasks. Furthermore, Multi-Residual Blocks [15] is similar to Multipath-DenseNet. However, a Multipath-denseblock is different from Multi-Residual blocks in the following ways: a)Multipath-denseblock uses concatenation, instead of summation in the residual connection; b) input is processed by a non-residual supervised transformation block before forwarding the feature maps to the next block; c) all layers of a Multipath-denseblock have a direct connection with a concatenating $1 \times 1$ convolutional layer in a supervised block which learns the drop paths of the network.

## 3. Methodology

The Multipath-DenseNet architecture is based on network ensemble [13] and DenseNet-denseblocks [14]. These architectures are discussed in detail in the following subsections.

### 3.1. Network ensemble

Deeper layers do not contribute to gradient propagation. Rather, they behave like ensembles of the same network [13]. Consider a residual network with 3 layers where information flows from input $x_0$ to output $y_3$

$$\begin{aligned} z_3 &= z_2 + f_3(z_2) \\ &= [z_1 + f_2(z_1)] + f_3[z_1 + f_2(z_1)] \\ &= [z_0 + f_1(z_0) + f_1[z_0 + f_1(z_0)]] \\ &\quad + f_3(z_0 + f_1(z_0) + f_1[z_1 + f_1(z_0)]) \end{aligned} \tag{1}$$

where $z$ represents the residual output of the layer and $f$ represents the residual function of the internal layers. Every path in a network behaves like a unique structure which decides whether to compute the function or skip it. These paths can be termed as a binary code which is 1 when the path with function is chosen, and 0 if it is skipped. Therefore, the network will have $2^n$ number of paths where $n$ represents the number of residual blocks.

In classical deep networks, a layer is dependent on only the output of its previous layer. Residual networks are different in this context; a residual function in ResNet receives input from $2^{n-1}$ different distributions generated from every possible configuration of the previous $n-1$ residual functions, confirming the fact that the ResNet make paths to carry the gradient information to the deeper layers.

Shallow paths contribute more to gradient magnitude than deeper paths, which means that effective paths in the residual network are relatively shallower. Moreover, paths in the ResNets are not of the same length i.e 1) there is one path which starts from an input and on the way to output, it passes through multiple functions. 2) there are $n$ number of paths that just pass through only one function. Gradient flows over these paths in the ResNet; however, all paths do not carry the same amount of the gradient. This implies that the path length affects the gradient magnitude. Gradient magnitude contributed by each length can be calculated by multiplying a similar path length's frequency by the expected gradient magnitude. This information can be used to create effective shallow paths in residual networks. The performance of the model trained on the effective paths is comparable to that of the full residual network [13].

### 3.2. Densenet-denseblocks

DenseNet improved the information flow between layers by directly connecting each layer to all subsequent layers and concatenating the feature maps. For example, feature maps received as input by $n$th layer in a $m$-layer network are: To simplify the notations, we use Haskell's operator symbol "++" for concatenation.

$$x_n = F_n(x_0 +\!+ x_1 +\!+ x_2 +\!+ \ldots +\!+ x_{n-1}) \tag{2}$$

where $(x_0 +\!+ x_1 +\!+ x_2 +\!+ \ldots +\!+ x_{n-1})$ is the concatenated output of the preceding layers and $F$ is a composite residual function [14]. This function is composed of three different operations which are batch normalization (BN), activation (ReLU), and convolution (CONV). The architecture of Densenet-denseblock is explained in Fig. 1. The authors in [11] proposed the architecture of a layer in the order of functions listed as BN-ReLU-CONV. Each layer in a denseblock will generate $k$ number of features, which is known as the growth rate. A small value of $k$ is selected, as the information is going to be concatenated in subsequent layers [14]. Concatenation operation is not valid when feature maps are reduced by pooling layers. Therefore, only the features of the same spatial size layers are concatenated.

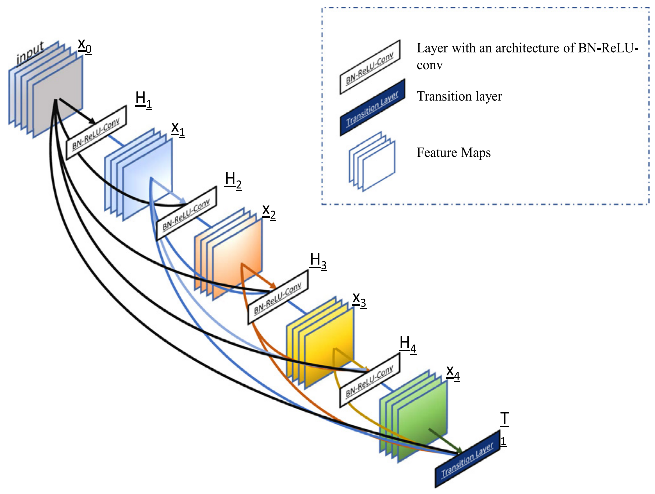

**Fig. 1.** A 5-layer dense block with a growth rate of k=4, showing that each layer is connected to all preceding layers. $x_0$, $x_1$, $x_2$, $x_3$, and $x_4$ represent the feature maps generated by the layers $H_1$, $H_2$, $H_3$, and $H_4$, respectively. The transition layer is represented as $T_1$. Architecture with three layers: batch normalization, rectified linear unit, and convolution (BN-ReLU-conv).

### 3.3. Multipath densenet

We propose a new architecture called Multipath-DenseNet which has more shorter and medium paths. The architecture of a Multipath-DenseNet is both deep and wide. We increase the width of the network by adding parallel dense blocks with similar depths. The architecture of Multipath-DenseNet is faster than a densely connected convolutional neural network. A block in Multipath-DenseNet contains $n$ number of denseblocks with a depth of $m$ and growth rate of $k$. The scale and depth of every parallel layer in each dense block in Multipath-Denseblock are the same. Every layer in a dense block of Multipath-Denseblock processes information of the same scale and depth level. At the end of every dense block, there is a supervised feature transformation block which learns intermediate features from the layers of these independent dense blocks. Input to the first convolutional layer of the transformation block can be formulated as below. To simplify the notations, we use Haskell's operator symbol "$++$" for concatenation.

$$Z_i = \{X_0 ++ X_1 ++ X_2 ++ \ldots ++ X_n\} \tag{3}$$

where $i$ is the $i^{th}$ Multipath-denseblock, $X$ represents the output from a denseblock, and $n$ is the total number of denseblocks. $X$ is defined as:

$$X_j = F_m\Big(x_0 ++ x_1 ++ x_2 ++ \ldots ++ x_{m-1}\Big) ++ \Big(x_0 ++ x_1 ++ x_2 ++ \ldots ++ x_{m-1}\Big) \tag{4}$$

where $j$ represents the $j$th DenseNet denseblock, while $x$ and $m$ represent a layer and the depth of the block, respectively. The size of the Multipath-denseblock is determined by the depth and width of the denseblock; depth represents the depth of a single independent shallow network and width represents the number of shallow networks in a block. In a traditional multi-residual network where the width is 2 and $f$ represents the residual function, the gradient will have the following four choices: 1) skip residual function $f1$; 2) skip $f2$; 3) skip both $f1$ and $f2$; and 4) take both $f1$ and $f2$. Therefore, the multiplicity of the network is defined as $2^{nk}$ where $n$ is the number of residual blocks and $k$ is the width of the residual block. Our network has many more residual connections which reach the output of a block. The multiplicity of Multipath-DenseNet is given as:

$$\begin{array}{ll} 2^{nk} + k(n-1) + 2n^2 & if \quad n > 1 \\ 2^{nk} & if \quad n = 1 \end{array} \tag{5}$$

Multipath-DenseNet can choose to skip the whole denseblock or a part of it, or take all of the parallel denseblocks because of the direct connection between the input and a supervision block. When the number of blocks is set to one, the number of paths is equal to the number of denseblocks in DenseNet. However, if the number of denseblocks is greater than one, the number of traversing paths increases in the network. Our proposed architecture Multipath-DenseNet converts longer traversing paths into shorter paths. A convolutional layer is added to the beginning of the network. Multipath-DenseNet generates twice more feature maps in this convolutional layer. Fig. 2 illustrates the architecture of Multipath-DenseNet with $n$ number of denseblocks.

#### 3.3.1. Supervised feature transformation block

The supervised feature transformation block consists of 3 composite layers with $1 \times 1$, $3 \times 3$, and $1 \times 1$ convolutional filters, respectively. The main aim of this supervised feature transformation block is to filter information and learn interme-

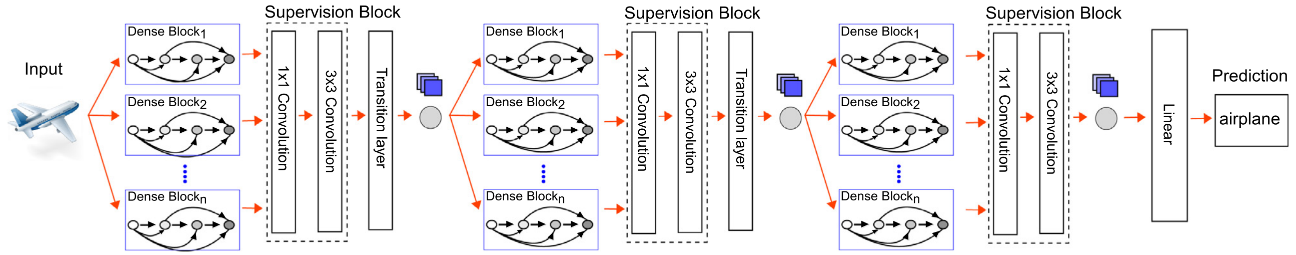

**Fig. 2.** Multipath-DenseNet architecture. Multipath-Denseblock consists of multiple denseblocks and a supervision block. The transition layers decrease the spatial size of the input. The linear layer represents the fully connected neural network layer for classification.

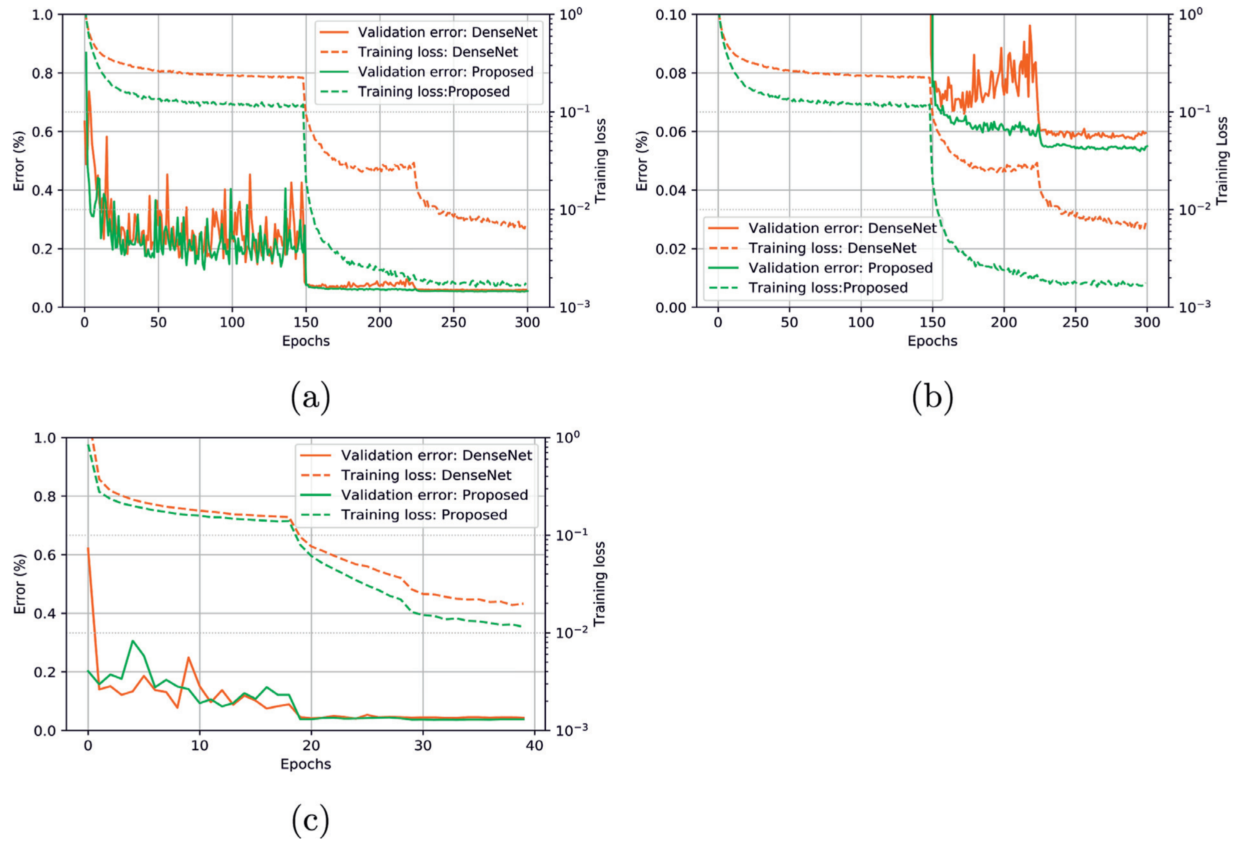

**Fig. 3.** *a*: Evolution of the training loss and validation error rate (%) of DenseNet-100 ($k = 12$) and Multipath-DenseNet (width = 5, depth = 6) ($k = 24$) on CIFAR-10. *b*: Enhanced view of Fig. 3a, highlighting on the drop in the error rate. *c* training loss and validation error comparison of DenseNet and Multipath-DenseNet on the SVHN dataset (without the extra training set).

diate features. The outputs of all independent denseblocks are not simply forwarded to the transition layer, as they are in DenseNet. Instead, the outputs are forwarded to a 1 × 1 convolutional layer. Due to the fact that the 1 × 1 layer is a single layer neural network, it learns to find an effective path. An effective path may be comprised of layers from multiple dense blocks. Moreover, the 1 × 1 layer reduces the number of features. The supervised feature transformation block also adds regularization to the network, which helps avoid overfitting. We concatenate the output of each denseblock in Multipath-Denseblock. This concatenated output is processed by a 1 × 1 convolutional layer of the supervised transformation block. The second layer is a non-residual convolutional layer, as it learns new features without previous information.

### *3.3.2. Pooling layer*

Instead of applying average pooling between the blocks, we used the convolution with stride 2.

### *3.3.3. Bottleneck layer with compression*

The bottleneck architecture improves the computational complexity by adding a $1 \times 1$ convolutional layer to every $3 \times 3$ convolutional layer [11]. The $1 \times 1$ convolutional layer reduces the number of input feature maps which will be processed by the preceding $3 \times 3$ layer. The bottleneck architecture of Multipath-DenseNet is referred as Multipath DenseNet-BC in the paper.

### *3.4. Implementation details*

We implemented Multipath-DenseNet using TensorFlow. The baseline experiments were conducted by the recommended publicly available TensorFlow implementation of DenseNet [14].

## **4. Experiments**

The efficiency of the Multipath-DenseNet architecture is evaluated on several benchmark datasets including CIFAR-10, CIFAR-100, Street View House Numbers (SVHN), and ImageNet. We tested the Multipath-DenseNet using various block sizes and growth rates.

### 4.1. Dataset

CIFAR-10 and CIFAR-100 are two labeled image datasets [21]. They each contain 60,000 images (50,000 for training and 10,000 for testing). The CIFAR-10 and CIFAR-100 datasets have 10 and 100 classes, respectively. Both of the datasets have images with a spatial size of 32 $\times$ 32, and the dominant object in a scene is the class object. The street view house numbers (SVHN) dataset [22] is a real-world image dataset developed for image recognition tasks. It is a colored image dataset and spatial dimensions of images are 32 $\times$ 32. It contains 73,256 images for training, 26,032 images for validation, and 531,131 images for extra training. The extra training set is a collection of easy tasks and gives benefits to some easy samples for recognition. Images were normalized by zero-mean as a preprocessing step. In the current implementation, we do not use the additional training due to resource constraints.

ImageNet [16] is a large-scale crowd-sourced annotated image dataset. It contains roughly 1.2 million images with over 1000 class labels for training and 50,000 images for validation. Due to its computational complexity, we down-sample the images to a spatial size of 32 $\times$ 32.

### 4.2. Augmentation

Data augmentation benefits the training process and reduces the use of regularization in the network. We adopted a widely employed data augmentation technique which is also used in [12,14,20]. Images were zero padded with 4 pixels on each side and then randomly cropped to the 32x32 original image size. Half of the images were horizontally mirrored. The augmented data is denoted with a âæ+âg sign. We used both CIFAR-10 and CIFAR-100 with their augmented datasets CIFAR-10+ and CIFAR-100+, respectively. We did not use any augmentation techniques for the SVHN or ImageNet datasets.

### 4.3. Training setup

Multipath-DenseNet was trained using the Stochastic gradient descent (SGD) method. We trained Multipath-DenseNet on the CIFAR dataset for 300 epochs and 40 epochs for SVHN and ImageNet dataset each. The hyper-parameter of SGD (learning rate) was set to 0.1 and the learning rate was reduced by the factor of 10 at 50% and 75% of training epochs. The dropout rate was set to 0.2. Nesterov momentum and weight decay were set to 0.9 and $10^{-4}$, respectively. We removed 0.1% of the training images from the CIFAR dataset and use them for validation. Also, 3000 images were selected randomly from the training set of the SVHN dataset for validation. The number of Multipath-denseblocks is set to 3 in the current implementation. Moreover, all the denseblocks of Multipath-denseblock have similar hyper parameter values.

## 5. Results and discussion

The results show that Multipath-DenseNet achieves similar or higher accuracy with fewer parameters and less memory. The parallel denseblocks increase the number of shorter and medium paths in Multipath-DenseNet, which allows for better gradient flow. Converting the very long residual connection into multiple short and medium connections reduces the memory and computational requirements. We compare the performance of Multipath-DenseNet with that of state-of-the-art DenseNet. DenseNet requires more memory than Multipath-DenseNet for a similar number of parameters and batch size (64) (Fig. 4a). In DenseNet, N number of feature copies are needed in a denseblock of size N. Therefore, DenseNet is memory inefficient and requires more time for learning tasks. We also measured the floating point operations (FLOPS) as a function of parameter size (Fig. 4c), which confirms that the Multipath architecture achieves similar accuracy with less computation and fewer parameters.Tables 1–3 show the error rate comparison of the CIFAR, ImageNet, and SVHN datasets, respectively.

### 5.1. Classification results on CIFAR

Multipath-DenseNet with a block size of (3,7) and a growth rate of 24 achieved an error rate of 5.78% (Table 1), and with a block size of (5,6) it achieved an error rate of 5.50% (Table 1), which are comparable to the error rates obtained by DenseNet. Multipath-DenseNet achieved the error rate of 5.50% (Table 1) with only 9.2M parameters, whereas DenseNet-100 achieved its rate with 27.2M parameters (Fig. 4b). The results shown in Tables 1 and 3 support the hypothesis that shorter connections contribute to gradient propagation. Baseline experiments of the DenseNet converged to slightly different error rates as mentioned in the original paper [14]. DesneNet-40 ($k = 12$) achieved an error rate of 5.59% and DenseNet-100 ($k = 12$) obtained an error rate of 4.86% on CIFAR-10+, while Multipath-DenseNet {5,6} ($k = 24$) obtained a comparable error rate of 4.57%. The second run of the baseline DenseNet-100 ($k = 12$) converged to the error rate of 6.5% and 25.2% on CIFAR-10 and CIFIAR-100, respectively, which are much higher than the previously obtained error rate.

### 5.2. Classification results on SVHN

Table 3 shows the performance of Multipath-DenseNet on the SVHN dataset. As explained earlier, we have used the SVHN dataset without its extra training set. Multipath-DenseNet achieved an error rate of 3.17% after 40 epochs while DenseNet-100($k = 12$) obtained an error rate of 7.24%. Fig. 3c shows the comparison of training and validation error rates convergence in DenseNet and Multipath-DenseNet.

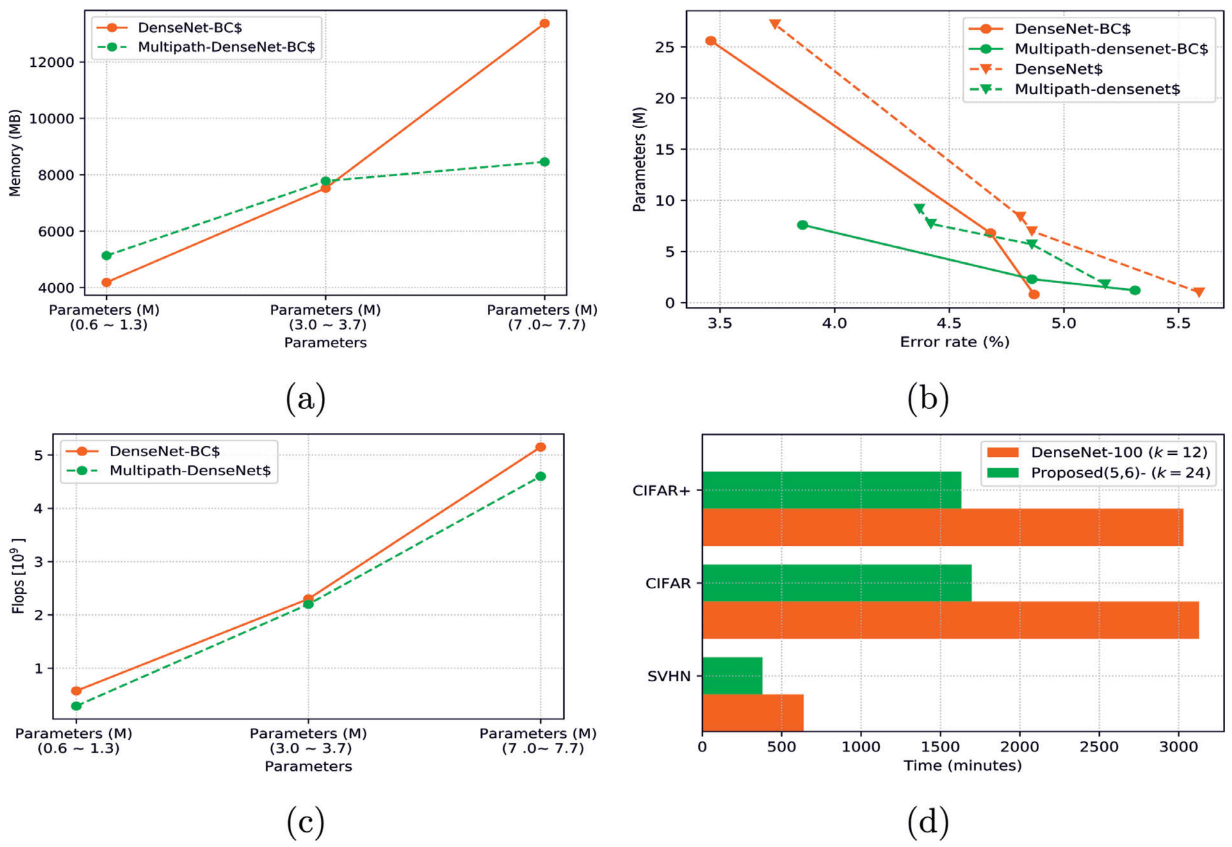

**Fig. 4.** *a (left)*: Memory requirements of DenseNet-BC and Multipath-DenseNet-BC. Multipath-DenseNet-BC requires less memory than DenseNet. *b*: Validation error rate comparison between DenseNet and Multipath-DenseNet with respect to the number of parameters. With fewer parameters, Multipath DenseNet achieves accuracy similar to that of DenseNet. *c*: Floating point operations (FLOPs) comparison between DenseNet and Multipath-DenseNet at the testing phase with respect to the number of parameters. *d*: Comparison of training completion times of DenseNet-100 ($k = 12$) and Multipath-DenseNet (width = 5, depth = 6) ($k = 24$) on CIFAR and SVHN.

### 5.3. *Classification on imagenet*

As shown in Table 2, Multipath-DenseNet achieved a 41.39% top-1 validation error with 6.4M parameters and without any augmentation or preprocessing, while the baseline algorithm [25] achieved 49.64% (without augmentation) and 40.96% (with augmentation) error rate with 37.1M parameters. AlexNet [5] achieved 40.7% top-1 validation error on the full-sized ImageNet dataset which has roughly 50 times more pixels per image. DenseNet [14] achieves an error rate of 22.15% on the full-sized ImageNet dataset. A higher image resolution yields better performance results [25].

### 5.4. *Discussion*

Table 1 shows that as we increase the number of ensemble networks or the denseblock size, the accuracy increases. This confirms that the Multipath-DenseNet is not overfitting the data and mitigates the complexities of DenseNet. Multipath-DenseNet outperforms DenseNet-100($k = 12$) on the SVHN dataset and the difference between the accuracies are much higher than CIFAR. This implies that DenseNet face difficulties in generalization because it requires more samples to handle complex tasks by its low-level features. Analysis on the training weights of DenseNet and Multipath-Densenet is given in the supplementary material. This phenomena is also explained in the next section.

#### 5.4.1. *Confidence score*

Features are roughly classified into low-level and high-level features. Low-level features can be extracted directly from images while high-level features depend on low-level features. In a DenseNet block, every layer receives input from all preceding layers, where low-level features dominate over detailed or high-level features. Deeper layers in DenseNet tend to make multiple smaller connections, as shown in [14]. Earlier layers of DenseNet, where the gradient information is reliable, tend to depend on longer connections but later layers depend mostly on multiple smaller connections in the network. This shows that DenseNet relies mostly on low-level features and it extracts low-level features through the deeper layers of

**Table 1**
Comparison of error rates (%) on the CIFAR-10 and CIFAR-100 datasets. + indicates that standard data augmentation was applied to the dataset. * indicates the mean result of three runs by us. With less memory and fewer parameters, Multipath DenseNet achieves comparable results as compared to DenseNet.

| Method | Depth | Width | Param(M) | C10(%) | C10+(%) | C100(%) | C100+(%) |
|---|---|---|---|---|---|---|---|
| Network in network [23] | – | – | – | 10.41 | 8.81 | 35.68 | – |
| Deeply supervised net [24] | – | – | – | 9.69 | 7.69 | – | 34.57 |
| ResNet [7] | 110 | – | 1.7 | – | 6.43 | – | 25.16 |
| ResNet with stochastic depth [10] | 110 | – | 1.7 | 11.6 | 5.23 | 37.8 | 24.58 |
| | 1202 | – | 10.2 | – | 4.91 | – | – |
| Wide ResNet [11] | 16 | – | 11 | – | 4.81 | – | 22.07 |
| | 28 | – | 36.5 | – | 4.17 | – | 20.5 |
| | 16 | – | 2.7 | – | – | – | – |
| DenseNet $k=12$ [14] | 40 | – | 1 | 6.67* | 5.59* | 27.83* | 25.62* |
| $k=12$ | 100 | – | 7 | 5.93* | 4.86* | 24.63* | 20.85* |
| $k=24$ | 100 | – | 27.2 | 5.83 | 3.74 | 23.42 | 19.25 |
| DenseNet-BC (k=12) | 100 | – | 0.8 | 6.21* | 4.87* | 25.01* | 23.12* |
| $k=24$ | 160 | – | 6.8* | 6.15* | 4.68* | 24.73* | 22.58* |
| $k=40$ | 190 | – | 25.6 | – | 3.46 | – | 17.18 |
| Multi-Residual Network [15] | 200 | 5 | 10.2 | – | 4.35 | – | 20.42 |
| | 398 | 5 | 20.4 | – | 3.92 | – | 20.59 |
| | 26 | 2 | 145 | – | 3.96 | – | 19.45 |
| | 26 | 4 | – | – | 3.73 | – | 19.6 |
| Multipath-DenseNet $k=12$) | 19 | 2 | 1.8 | 6.74 | 5.18 | 22.78 | – |
| $k=12$ | 19 | 7 | 3.5 | 6.0 | 4.86 | 22.36 | 22.53 |
| $k=24$ | 19 | 7 | 5.7 | 5.78 | 4.42 | 22.49 | 22.76 |
| $k=24$ | 19 | 12 | 7.7 | 5.18 | – | 22.53 | 22.03 |
| $k=24$ | 25 | 6 | 9.2 | 5.5 | 4.57 | 21.98 | 21.00 |
| Multipath-DenseNet-BC $k=12$ | 19 | 6 | 1.2 | 6.23 | 5.31 | 22.4 | 23.96 |
| $k=32$ | 19 | 14 | 7.6 | 5.23 | 3.86 | 22.1 | 19.70 |

**Table 2**
ImageNet (resized to $32 \times 32$) top-1 validation error rate (%).

| Method | Error rate (%) | Time (days) |
|---|---|---|
| DenseNet-BC-100 ($k=12$) | 46.63 | 5.2 |
| Wide-ResNet 28-10 | 49.64 | 7.6 |
| Multipath-DenseNet -{2,8}($k=32$) | 41.39 | 2.3 |

**Table 3**
SVHN (without the extra training set) error rate (%) comparison.

| Method | Error rate (%) |
|---|---|
| DenseNet-100 ($k=12$) | 7.24 |
| Multipath-DenseNet -{5,6}($k=24$) | 3.17 |

DenseNet-denseblock. We noticed a trend of increase in loss after fewer number of epochs with the decrease in accuracy at different configurations of DenseNet (Fig. 3b is an example of one configuration). One of the possible reason for this trend is that the network overfits the data and the decrease in accuracy indicates that model is unable to recover by the regularization techniques. At this point, learning rate decrement will only trap the network in its current performance surface. This effect is also caused when a network is not "confident" about its prediction and it generates borderline probabilities. The behavior which is explained earlier restricts the generalization capability of a model on complex classification tasks. DenseNet's performance on the SVHN dataset without the extra training set, and its performance on CIFAR-10, as compared with CIFAR-100, confirm that DenseNet tends to overfit the easy tasks as the high level features dominate the DenseNet architecture. We also plot the weights of two DenseNet models as heatmaps. The DenseNet models are trained on the SVHN and CIFAR-10 datasets, respectively. The heatmaps of the DenseNet models are presented in Figs. S1 and S2, respectively, in the supplementary file. The heatmaps show dependence of a layer on its preceding layers in DenseNet. In Figs. S1 and S2, while tracking paths from the last layer to the first layer of a denseblock, we find a set of shorter paths. This shows that DenseNet heavily relies on its initial layers. We also trained Multipath-DenseNet with a width of 5 and height of 6 and show its heatmap of weights in figure S3 in the supplementary file.

### 5.4.2. Improved training time

Multipath-DenseNet substantially reduces the training time by limiting the feature size in a Supervised Feature Transformation Block. A very deep DenseNet suffers from the curse of dimensionality due to high-dimensional feature maps. Applying convolutional operation in high-dimensional feature maps becomes computationally expensive. Multipath-DenseNet allows using multiple GPUs for independent multi-blocks. As shown in Fig. 4 d, Multipath-DenseNet{5,6} ($k = 24$) reduced the training time by approximately 50%, compared with DenseNet-100($k = 12$). The given training time was calculated on the resource having Xeon(R) E5-2630 v4@2.2Ghz with Titan X (Pascal) GPU (12GB memory).

### 5.4.3. Deep supervision

Deep supervision improves network learning by applying an auxiliary classifier to the intermediate layers of a network for faster and efficient training. We hypothesize that the Multipath-DenseNet structure behaves similar to that of deep supervision [26] by supervised feature transformation block in the Multipath-DenseNet architecture. However, a single loss function is shared by all the layers in Multipath-DenseNet. We applied Multipath-DenseNet without the deep supervision block, which does not achieve comparable results. The error rate of Multipath-DenseNet {5,6} ($k = 24$) increased from 4.57% to 11.38%.

## 6. Conclusion

In this paper, we presented Multipath-DenseNet, a robust and efficient block architecture which uses independent ensemble denseblocks to learn shorter neural connections. With a smaller number of parameters, Multipath-DenseNet achieves higher accuracy than other baselines. The experimental results of this study demonstrate that shorter and longer connections both are crucial for better learning of deep neural networks. Multipath-DenseNet keeps the focus on both long and short connections and improves performance on various image classification datasets.

## Acknowledgments

This research was supported by the National Research Foundation of Korea (No. 2017R1A2A1A17069645).

## Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ins.2019.01.012.

## References

[1] P.O. Pinheiro, R. Collobert, P. Dollár, Learning to segment object candidates, in: Advances in Neural Information Processing Systems, 2015, pp. 1990–1998.
[2] R. Collobert, J. Weston, 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, pp. 160–167.
[3] A. Van den Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, in: Advances in Neural Information Processing Systems, 2013, pp. 2643–2651.
[4] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324, doi:10.1109/5.726791.
[5] P.L. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a Meeting Held December 3–6, 2012, Lake Tahoe, Nevada, United States, 2012.
[6] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR, abs/1409.1556 (2014).
[7] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, CoRR, abs/1512.03385 (2015).
[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S.E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, CoRR, abs/1409.4842 (2014).
[9] C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7–12, 2015, Montreal, Quebec, Canada, 2015.
[10] G. Huang, Y. Sun, Z. Liu, D. Sedra, K.Q. Weinberger, Deep networks with stochastic depth, CoRR, abs/1603.09382 (2016).
[11] R.C. Wilson, E.R. Hancock, W.A.P. Smith (Eds.), Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19–22, 2016, BMVA Press, 2016.
[12] G. Larsson, M. Maire, G. Shakhnarovich, Fractalnet: ultra-deep neural networks without residuals, CoRR, abs/1605.07648 (2016).
[13] A. Veit, M.J. Wilber, S.J. Belongie, Residual networks are exponential ensembles of relatively shallow networks, CoRR, abs/1605.06431 (2016).
[14] G. Huang, Z. Liu, K.Q. Weinberger, Densely connected convolutional networks, CoRR, abs/1608.06993 (2016).
[15] M. Abdi, S. Nahavandi, Multi-residual networks, CoRR, abs/1609.05672 (2016).
[16] 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20–25 June 2009, Miami, Florida, USA, IEEE Computer Society, 2009.
[17] K. He, J. Sun, Convolutional neural networks at constrained time cost, CoRR, abs/1412.1710 (2014).
[18] Y.W. Teh, D.M. Titterington (Eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13–15, 2010, 9, JMLR.org, 2010. JMLR Proceedings
[19] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[20] S. Xie, R.B. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, CoRR, abs/1611.05431 (2016).
[21] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images (2009).
[22] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A.Y. Ng, Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011, 2011, p. 5.
[23] M. Lin, Q. Chen, S. Yan, Network in network, arXiv:1312.4400 (2013).
[24] G. Lebanon, S.V.N. Vishwanathan (Eds.), Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9–12, 2015, 38, JMLR.org, 2015. JMLR Workshop and Conference Proceedings
[25] P. Chrabaszcz, I. Loshchilov, F. Hutter, A downsampled variant of imagenet as an alternative to the CIFAR datasets, CoRR, abs/1707.08819 (2017).
[26] L. Wang, C. Lee, Z. Tu, S. Lazebnik, Training deeper convolutional networks with deep supervision, CoRR, abs/1505.02496 (2015).