In this letter, we propose a sequential convolutional residual network, where we first analyze a tangled network architecture using simplified equations and determine the critical point to untangle the complex network architecture. Although the residual network shows good performance, the learning efficiency is not better than expected at deeper layers because the network is excessively intertwined. To solve this problem, we propose a network in which the information is transmitted sequentially. In this architecture, the neighboring layer output adds the input of the current layer and iteratively passes its result to the next sequential layer. Thus, the proposed network can improve the learning efficiency and performance by successfully mitigating the complexity in deep networks. We show that the proposed network performs well on the Cifar-10 and Cifar-100 datasets. In particular, we prove that the proposed method is superior to the baseline method as the depth increases.

**key words:** deep learning, convolutional neural network, residual network

### 1. Introduction

Image recognition studies are moving from hand-crafted features to deep learning, and convolutional neural networks [1]–[4] with deep layers have recently achieved the best accuracy for image recognition. Specifically, Simonyan et al. [1] proposed a deep convolutional network using 3 $\times$ 3 fixed filters in a 19 layer-based architecture; however, they failed to stack more layers because of the vanishing-gradient problem. To address this issue, He et al. [2] proposed a residual network (ResNet) with an identity shortcut connection and managed to stack more than 1,000 layers. Furthermore, in [3], they investigated the propagation formulations behind the residual connection and modified the ResNet architecture using identity skip connections to achieve better accuracy. On the other hand, Zagoruyko et al. [4] proposed a wide residual-network architecture (WideNet) where the 16 layer-based wide network outperformed the 100 layer-based deep network, where they have led the fast training of the network and overcome a problem of diminishing feature reuse. In this respect, many studies try to stack deeper or wider layers to make complex network architectures. Recently, DenseNet [5] was introduced for connecting each layer to every other layer in a feed-forward fashion and it achieved the best accuracy but its parameter size is 27.2 M in 100 layer-based network. When analyzing the ResNet style network architecture, ResNet basically has better performance when stacking many layers; however, its performance is saturated when it exceeds a certain number of layers. For example, in [2], a 56 layer (0.85 M parameter) ResNet achieved a 6.97% error rate in the Cifar-10 dataset [6], while a 110 layer (1.7 M parameter) ResNet showed only a 6.43% error rate. The parameters doubled but the performance did not increase correspondingly. As Veit et al. [7] argued, the deep ResNet acts like ensembles of relatively shallow networks. Xie et al. [8] constructed a network by repeating a building block that aggregated a set of transformations with the same topology. From the viewpoint of the basic ResNet architecture, very tangled network architectures do not always guarantee the best performance.

In this letter, we propose a sequential ResNet for efficiently untangling the complex network architecture. First, we designed a parallelized ResNet, simply to solve the tangled network architecture, and compared it with the original ResNet, which could be considered a serialized ResNet. After deriving simpler equations for the ResNet architecture, we realized that the parallelized ResNet is too simple to show good performance. Therefore, we designed a novel network architecture where the neighboring layers are connected, but the distant layers are not. We add the output of the previous layer to the input of the current layer. We call it the Sequential ResNet because the image goes through the neighboring layers sequentially. Note that GoogLeNet [9] split the input into convolutional layers using the different kernel sizes and then merged them. On the other hand, the proposed method does not split the input for the different kernels, but for the sequential deployed layers.

### 2. Sequential Residual Network

Deep residual networks have a serialized stack of layers, as shown in Fig. 1 (a); it can be expressed with the following equation:

$$x_{i+1} = x_i + f_i(x_i),$$

where $f_i(x_i)$ includes the convolutional layers, and $x_i$ and $x_{i+1}$ are an input and an output, respectively. As shown in Eq. (1), the identical connection, $x_i$, could reduce the vanishing-gradient issue in the deep network. In this letter, to efficiently interpret the network architecture, we ignore the Rectified Linear Unit (ReLU) and Batch Normalization (BN) notations in $f_i$. Now, we can derive the following simple equations:

[Manuscript received October 23, 2017.]
[Manuscript revised December 14, 2017.]
[Manuscript publicized January 18, 2018.]

The author is with the Department of Software and Computer Engineering, Ajou University, Republic of Korea.

\( ^{†} \)E-mail: wjhwang@ajou.ac.kr (Corresponding author)

DOI: 10.1587/transinf.2017EDL8233
where the main difference between Eq. (4) and Eq. (6) is \( f_j(f_i(x_1)) \); there are only \( f_2(f_i(x_1)) \) and \( f_3(f_i(x_1)) \) terms. Thus, we call the proposed method a sequential ResNet because the proposed architecture consists of sequential convolutional layers. In detail, the \( f_i \) module is only correlated with neighboring modules, e.g., \( f_{i-1} \) and \( f_{i+1} \). There is no connection to the remote modules, which means that the proposed method has a slightly less tangled network and we have more chances to efficiently learn deeper networks. The detailed network architectures are compared in Fig. 2 in deeper network architectures.

3. Experimental Results and Discussion

We evaluated our method using the Cifar-10 and Cifar-100 datasets [6]. Cifar-10 and Cifar-100 consist of 50,000 training images and 10,000 test images with 10 classes and 100 classes, respectively; all images are 32×32 pixel-based color images. For data augmentation, we do horizontal flips and random crops with four pixels’ padding on each side. The image is normalized by the color variances and means. The network implementation details are as follows: The deep learning models are trained using the Caffe deep-learning framework and eight Titan X GPUs. The weight decay is 0.001 and the momentum is 0.9. The mini-batch size is 256 and the learning rate started with 0.1 and was divided by 10 at the 32-k and 48-k iterations. The model had 64-k total iterations.

The performances of ResNet [2], WideNet [4], the parallelized network, and the proposed network are compared in Fig. 3. The widths of WideNet [4] are set to the same parameter sizes as ResNet. For example, the 20 layer-based ResNet has 3×2 repetitive convolutional layers with 16, 32, and 64 widths, as shown in Fig. 2. WideNet has 1×2 repetitive layers with 16×3, 32×3, and 64×3 widths, respectively.
Table 1 Classification error for the Cifar-10 test set. * means the method re-implemented using Caffe.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>8.75%</td>
<td>8.18%</td>
<td>8.09%</td>
<td>7.19%</td>
</tr>
<tr>
<td>32</td>
<td>7.51%</td>
<td>7.83%</td>
<td>7.28%</td>
<td>6.47%</td>
</tr>
<tr>
<td>62</td>
<td>-</td>
<td>7.37%</td>
<td>6.30%</td>
<td>5.65%</td>
</tr>
<tr>
<td>122</td>
<td>-</td>
<td>6.65%</td>
<td>6.53%</td>
<td>5.46%</td>
</tr>
<tr>
<td>182</td>
<td>-</td>
<td>6.67%</td>
<td>6.13%</td>
<td>5.16%</td>
</tr>
</tbody>
</table>

The WideNet showed slightly worse performance than the baseline ResNet with the same parameter sizes, and the parallelized network also achieved the worst accuracy due to no correlation between layers. The proposed method consistently showed the best performance, regardless of the number of layers.

Now, we compare the performance of the proposed method with the ResNet as the baseline method from 20 to 182 layers in Table 1. For example, the 20, 32, 62, 122, and 182 layer configurations have $3 \times 2$, $5 \times 2$, $10 \times 2$, $20 \times 2$, and $30 \times 2$ repetitive layers in the blocks of the same widths, respectively and there are three different widths (e.g., 16, 32, 64). For the various layer-based architecture comparisons, we re-implemented ResNet [2], [3] using Caffe. Note that we only built the architecture with the basic blocks, not the bottleneck block, because our main goal was to investigate the network efficiency, not build deepest layers. Table 1 confirms that the re-implemented ResNet shows results similar to [2] from 20 to 34 layers. In the Cifar-10 test protocol, the proposed method demonstrated better performance from 20 to 182 layers. From the viewpoint of increased performance according to the increased layers, the proposed method’s error rate improved 2.03% between 20 and 182 layers, while ResNet [2] only achieved a 1.51% improvement.

This tendency is also shown in Cifar-100 database as shown in Table 2; e.g., the baseline [2] improves by 3.45% but the proposed method improves by 7.22%. In this respect, we can infer the proposed method has a more efficient learning architecture for building deep networks because of the sequential layer deployment. Moreover, as shown in Table 2, the proposed method shows better performance compared with ResNet [2], [3] in deeper networks. For example, the ResNet archived 27.56% and 27.48% in 122 and 182 layers, respectively but the proposed network shows 25.89% and 25.12%, respectively.

4. Comparison with the State-of-the-Art Method

Table 3 shows the comparison results on Cifar-10 database in well-known deep networks such as ResNet [2], WideNet [4], DenseNet [5], and the proposed method. The best accuracy has been achieved by DenseNet but the DenseNet needs a huge amount of parameters for training a model, e.g., 27.2M, and the WideNet also has 11.0M parameters for achieving 4.81%. On the other hand, the proposed method shows better result than the original ResNet using only 17.M parameters. All in all, DenseNet has achieved the best accuracy from the viewpoint of the absolute performance without consideration of its parameter size, but the proposed method is efficient when considering its parameter size and accuracy in Cifar-10 database.

5. Conclusion

We proposed a sequential ResNet that considered only the correlation of the neighboring convolutional blocks. It improved the learning efficiency when the network was deeply stacked. From experimental results using the Cifar datasets, we confirmed that the proposed network architecture worked efficiently.

Acknowledgments

This work was supported by the new faculty research fund of Ajou University.

References


