arXiv:1611.07718v1 [cs.CV] 23 Nov 2016

On the Connection of Deep Fusion to Ensembling

Liming Zhao[1] Jingdong Wang[2] Xi Li[1] Zhuowen Tu[3] Wenjun Zeng[2]
[1]Zhejing University [2]Microsoft Research [3]UC San Diego
{zhaoliming,xilizju}@zju.edu.cn {jingdw,wezeng}@microsoft.com ztu@ucsd.edu

Abstract

In this paper, we provide a systematic study to the prevailing ResNet architecture by showing a connection from a general deeply-fused net view to ensembling. We start by empirically demonstrating the resemblance between the expanded form of the deeply-fused net and an ensemble of neural networks. Our empirical results uncover that the deepest network among the ensemble components does not contribute the most significantly to the overall performance and instead it provides a manner to introduce many layers and thus guarantee the ensemble size. Guided by the above study and observation, we develop a new deeply-fused network that combines two networks in a merge-and-run fusion manner. It is less deep than a ResNet with the same number of parameters but yields an ensemble of the same number of more-capable component networks, thus improving the classification accuracy. We evaluate the proposed network on the standard recognition tasks. Our approach demonstrates consistent improvements over the ResNet with the comparable setup, and achieves the state-of-the-art results (e.g., 3.57% testing error on CIFAR-10, 19.00% on CIFAR-100, 1.51% on SVHN).

1. Introduction

Deep convolutional neural network, since the breakthrough result in the ImageNet classification challenge [14], has been widely studied [31, 28, 8]. It has achieved surprising performance in many other computer vision tasks, including object detection [5], semantic segmentation [18], edge detection [37], and so on.

Recent developments in the deep network architecture design, such as ResNet [8], Highway [28], and GoogLeNet [31], have been attracting a lot of attention. The study in deep fusion [36], FractalNet [15], ensemble view [35] points out that such networks are a mixture of an exponential number of paths. One observation from [35] is that removing single residual layers from a ResNet at test time does not noticeably affect the performance, which is viewed as an evidence of the connection to ensembling.

In this paper, we are interested in the connection of ResNet and its generalized form, deeply fused net to an ensemble of networks. We study the connection from two perspectives. On the one hand, we study the architecture connection. A deeply fused net can be equivalently expanded to a tandem ensemble form, i.e., a combination of weight-shared paths (networks) with information communication across networks. The structure of a tandem ensemble resembles that of the conventional ensemble of weight-shared networks, but without information communication across component networks. The empirical results also show that the performances of the tandem ensemble (with ResNet-like deep fusion as examples) and the conventional ensemble are close. On the other hand, we study how the tandem ensemble performs when varying the ensemble size and the component network capabilities. It empirically shows similar behaviors with the conventional ensemble.

Furthermore, we study the role of the deepest component network in an ensemble of weight-shared networks. Our analysis suggests the deepest network essentially provides a way to include many layers and thus guarantee the ensemble size, while itself does not make the most significant contribution to the overall performance though with weight sharing the difficulty of training the deepest network is reduced. Inspired by the connection to ensembling and the effect of the deepest network, we present a merge-and-run fusion scheme to combine two networks, yielding a composite network in which each fusion unit corresponds to two fusion units in a deeper ResNet. Overall, the deeply merge-and-run fused network (DFN-MR) performs superiorly to the ResNet with the comparable setup on CIFAR-10, CIFAR-100, SVHN, and ImageNet, and achieves state-of-the-art performance: 3.57% testing error on CIFAR-10, 19.00% on CIFAR-100, 1.51% on SVHN.

In summary, the main contributions are listed as follows.

- We study the connection of ResNets and deep fusion, to ensembling and show the architecture and empirical performance resemblance.
- Our empirical results imply that the deepest network plays a role of including many layers to guaran-

tee the ensemble size, and that its improved performance, through sharing weights with shallower ensemble component networks, does not make the most significant contribution to the overall performance.

- Guided by the above points, we present a deeply merge-and-run fused network, which achieves state-of-the-art performances on CIFAR-10, CIFAR-100, and SVHN.

2. Related Work

There have been rapid and great progress of deep neural networks in various aspects, such as optimization techniques [29, 12, 19], initialization schemes [20], regularization strategies [27], activation and pooling functions [7, 3], network architecture [17, 23, 21, 25], and applications. In particular, recently network architecture design with the increasing number of layers has been attracting a lot of attention.

Highway networks [28], ResNets [8, 9], and GoogLeNet [30] are shown to be able to effectively train a very deep (over 40 and even hundreds or thousands) network. The identity connection or the bypass paths are thought as the key factor to make the training of very deep networks easy. Following the ResNet architecture, several variants are developed by modifying the architectures, such as wide residual networks [38], ResNet in ResNet [33], multilevel residual networks [39], multi-residual networks [1], and so on. Another variant, DenseNets [10], links all layers with an identity connection and is able to fully explore the potential of the network through feature reuse. In addition, optimization techniques, such as stochastic depth [11] for ResNet optimization, are developed.

Deeply-fused networks [36], FractalNet [15], and ensemble view [35] point out that a ResNet and a GoogleNet [31, 32, 30] are a mixture of many dependent networks. Ensemble view [35] observes that ResNets behave like an exponential ensemble of relatively shallow networks, and point out that introducing short paths helps ResNets to avoid the vanishing gradient problem, which is similar to the analysis in deeply-fused networks [36] and FractalNet [15] that additionally design new architectures.

In this paper, we study the connection to ensembling, and provide a comprehensive study complementary to the closely-related study in [35]. There are substantial differences between [35] and our work: (i) We show the architecture resemblance of the deeply-fused net and an ensemble of neural networks by expanding the deeply-fused net to a structure different from [35]. (ii) We show the classification performance resemblance and the performance variation resemblance by changing the ensemble size and the ensemble component capability. We directly *train* the composed networks and ensembles *from scratch* for comparing their performances, instead of removing single layers/modules, or reordering modules at *test time* done in [35]. (iii) Our study on how the training of the deep component networks is influenced by the shallow networks is also different from [35]. In particular, we suggest that the deepest (plain) network introduce many layers to generate a large ensemble. (iv) Finally, guided by our analysis, we present a novel deeply-fused network with merge-and-run fusion.

3. On the Connection to Ensembling

We describe the ResNet in the general deep fusion form. There are 2 base networks: One is deep $N^R : C \to B_1^R \to B_2^R \to \cdots \to B_K^R \to FC$; the other one is shallow $N^L : C \to B_1^L \to B_2^L \to \cdots \to B_K^L \to FC$. C is a shared layer taking the image as the input, and FC is a shared fully-connected layer for classification[1]. B_k^R is a block, composed of a serial of convolutional layers. B_k^L is an identity layer, or a convolutional layer that exists only when the corresponding block in the deep base network contains a down-scale operation. Deep fusion combines the blocks across 2 base networks: $\bar{\mathbf{x}}_k = H_k(B_k^L(\bar{\mathbf{x}}_{k-1}), B_k^R(\bar{\mathbf{x}}_{k-1}))$, where $\bar{\mathbf{x}}_0 = \mathbf{x}_0 = C(\mathbf{I})$, and H_k is a combinator, e.g., summation, maximum, or other forms. It can be generalized to more base networks and each block is not limited to an identity connection in the shallow base network. Figure 1(a), Figures 3(b), and 3(c) show three ResNet-like deeply fused networks.

In the following, we study the connection of ResNet-like deeply-fused networks to ensembling from the architecture and classification performance resemblance, and how the performance is affected when varying the ensemble size and the component network capabilities on an example dataset CIFAR-10. A few more studies, e.g., on ImageNet, are provided in Appendix A.1.

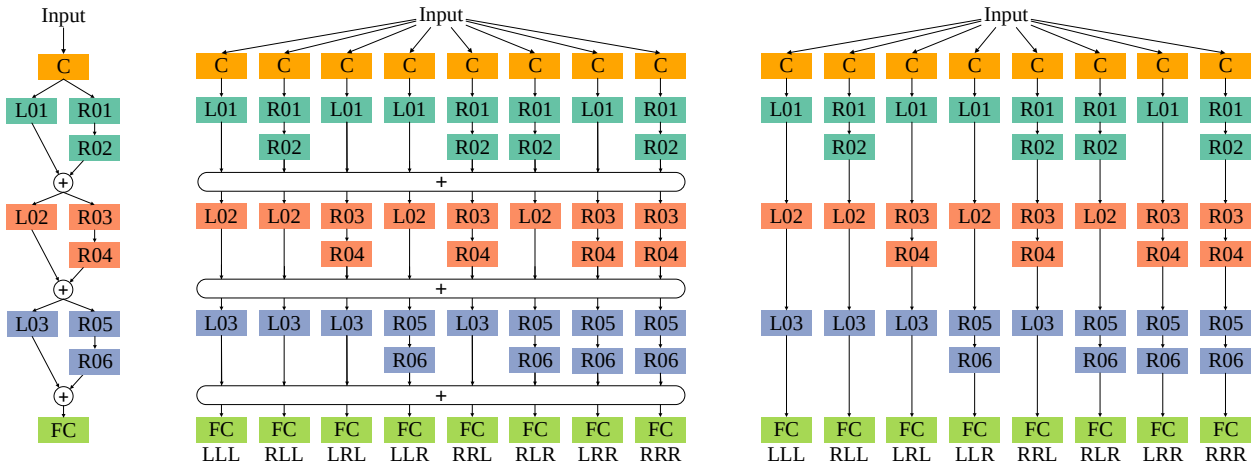
3.1. Architecture and Performance Resemblance

In the network composed from N^R and N^L, the information can be flowed from $\mathbf{x}_0$ to $\bar{\mathbf{x}}_K$ through many paths other than the two paths N^R and N^L. In each block k, there are two choices $\{B_k^R, B_k^L\}$, and thus there are totally 2^K paths from $\mathbf{x}_0$ to $\bar{\mathbf{x}}_K$: $\{B_1^{x_1} \to B_2^{x_2} \to \cdots \to B_K^{x_K}; x_k \in \{L, R\}\}$.

Consider the simple deeply-fused net N_f, shown in Figure 1(a), where there are $K = 3$ blocks. The 8 (2^3) information flow paths (networks) with weight sharing, denoted by N_e, are depicted in Figure 1(c). We expand the composite net N_f to an equivalent composite net N_{ce}[2], shown

[1] There may be other forms for the first convolutional layer and the classification layer, and it is possible that the two layers are included to blocks.

[2] This composite network N_{ce} itself is a deeply-fused net, which can be further expanded, like the expansion from N_f to N_{ce}, to a net N'_{ce}. It is easy to show that there are duplicate paths in N'_{ce} and removing the



(a) N_f (b) N_{ce} (c) N_e

Figure 1. Architecture resemblance. (a) A simple fused net; (b) The tandem ensemble; an expanded form of (a); and (c) an ensemble of the same components. The architecture structures of (b) and (c) are close except that there is information communication across component networks in (b). The colored boxes with R·· and L·· inside are layers. There are three blocks in the right path in (a): B_1^R =(R01, R02), B_2^R =(R03, R04), B_3^R =(R05, R06). The 8 combinations (e.g., LLL, RLL, ...) of L and R are used to name the 8 component networks.

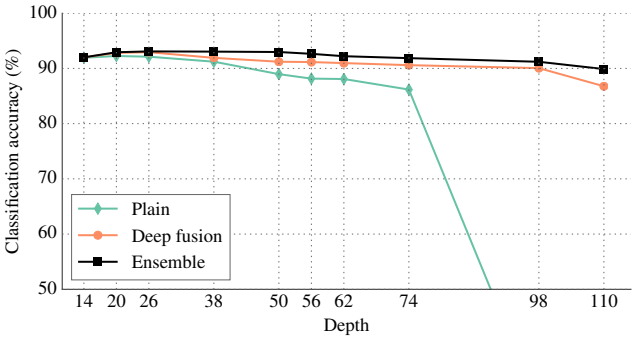


Figure 2. Empirical performance resemblance. It can be seen that both the DFN and the ensemble outperform the corresponding plain network with various depths, and they perform similarly.

in Figure 1(b). We can see that the 8 nets, N_e shown in Figure 1(c), if combined together, form an ensemble, where weights are shared across networks and no information is exchanged between 8 component networks in the forward process. In contrast, the network N_{ce} shown in Figure 1(b), which we call a tandem ensemble, is also composed of the same 8 component networks but with information exchange modules (fusion) across networks. *The architecture resemblance between the tandem ensemble (Figure 1(b)) and the ensemble (Figure 1(c)) suggests the close relation between the fused net and an ensemble*.

Furthermore, we compare the empirical performance between a deeply-fused net and an ensemble of the corresponding component networks. We take the fused nets with three ($K = 3$) fusions as examples (shown in Figure 3(b)), which are composed from a deep base network (also called plain network) with various depths, $14, 20, \ldots, 110$[3] and a 5-layer shallow base network.

The empirical results over CIFAR-10 are given in Figure 2 (without special specification, the later empirical analysis results are all reported over the augmented CIFAR-10 dataset). Both DFNs and ensembles of networks outperform the plain network, and *perform similarly, empirically suggesting their connection*. It is noticed that the ensemble performs better than DFNs, for the depth being over 20. The reason is that the diversity of a DFN is worse than the corresponding ensemble because of higher correlation between component networks which stems from information communication across component networks.

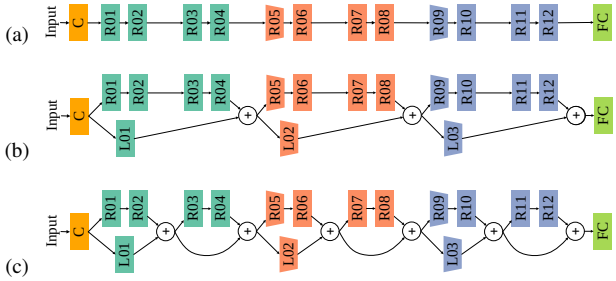


Figure 3. ResNet-like DFN examples. (a) A plain network, (b) A composed network with three fusions, (c) A composed network with six fusions. The trapezoid shape indicates that downsampling (stride=2) occurs. In (b) and (c), the shallow base network contains three convolutional layers for fusions besides the first convolutional layer C and the last layer FC, and identity layers if there are more than three fusions.

3.2. Performance Variation Resemblance

To make a further study on the connection, we empirically investigate how the number of component networks

duplicates yields N_{ce}, which is equivalent to N'_{ce}.

[3]The depth is in the form of $2 + x \times K$ for convenience, and x is the length of each block

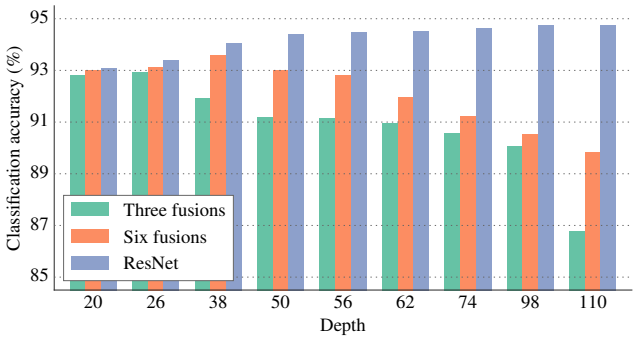


Figure 4. Performance variation when changing ensemble sizes for various depths. The ensemble size of N_{f3} is the smallest ($2^3 = 8$), that of N_{f6} is the medium ($2^6 = 64$), and that of a ResNet is the largest ($2^{\frac{depth-2}{2}}$, each block includes 2 layers). For each depth, *the DFN with a larger ensemble size performs better*: the ResNet performs the best and the DFN N_{f3} performs the worst, which is similar to the general property of the conventional ensemble.

(the ensemble size) and the capability of each individual network (a component classifier) influence the performance of the deeply-fused net.

Ensemble size. We conduct a simple study: Compare two DFNs composed from the same 5-layer shallow and deep networks, but with different numbers of fusions, which are illustrated in Figure 3. In the deeply-fused net denoted by N_{f3} with 3 fusions, shown in Figure 3(b), each fusion follows a one-layer branch and a x-layer branch (x is the number of layers in the branch), while in the net denoted by N_{f6} with 6 fusions, shown in Figure 3(c) each fusion follows an identity-layer or a one-layer branch and a $\frac{x}{2}$-layer branch. In our comparison, we vary the depth of the deep network forming the two DFNs from $20, 26, \ldots, 110$ (i.e., $x = 6, 8, \ldots, 36$). It is easy to verify that the ensemble sizes of N_{f3} and N_{f6} are 8 ($= 2^3$) and 64 ($= 2^6$), respectively and the 64 ensemble component networks from N_{f6} contain the 8 ensemble component networks from N_{f3}. We also report the results from ResNets with each fusion following a two-layer branch and another identity or one-layer branch, which leads to a larger ensemble size.

The comparison is given in Figure 4. It can be seen that DFNs N_{f6} (the orange bar) always outperform N_{f3} (the green bar) for various depths. This implies that *like the conventional ensemble, the deeply-fused nets with larger ensemble sizes perform better*[4]. We also observe that ResNets perform the best, which is because the ensemble size is the largest (though some component networks of a DFN may not be contained in those of a ResNet). Similar results are observed in [38] that more convolutional layers per block in ResNet (from 2 to 3 and 4) produce worse performance, since the ensemble size is reduced in the last two cases.

The superior performance of our proposed DFN with merge-and-run fusion over the DFN with inception-like fusion, shown in Table 4, also shows the consistent property, better performance with a larger ensemble size.

Component Capacity. We compare deeply-fused nets with the same ensemble size but different component networks. We consider DFNs with 3 fusions, like Figure 3(b), composed from plain networks with different depths, from $14, 20, 26, 38, 50, \ldots, 110$. The depths of their component networks are summarized in Table 1.

According to the empirical evidence shown in [8] and in Figure 2 that a network with the depth greater than roughly 20 layers becomes difficult to train, and thus its empirical capability becomes lower when the network is deeper (though its ideal capability could be higher), we predict the capability of each DFN by considering the overall capability of the component networks, and summarize them in the column (Pred. Rel. Cap.) of Table 1. The empirical results for DFNs as well as their corresponding ensembles (with weight sharing) are also reported in Table 1. The results are consistent to the prediction, which verifies that *like the conventional ensemble, a DFN, or a tandem ensemble of better component networks tends to perform better*[5].

4. On the Deepest Component Network

We empirically study how the training of the deepest network (i.e., the deep base (plain) network for forming the ResNet) are affected by weight sharing with shallow networks and how the deepest network contributes to the overall performance. We conduct the analysis over the conventional ensemble of weight-shared component networks, as we can have the performance for each individual component network in the conventional ensemble, while in the tandem ensemble the performance of each individual network is not so meaningful because the parameters of each component network are learnt with information communicated with other component networks, i.e., each component network does not form an independent classifier.

Weight sharing. We compare the performances between the ensembles of networks with and without weight sharing. It is shown in Figure 2 that when the depth of plain network is more than 20 training the network becomes difficult. So we consider greater depths, $26, 38, \cdots, 110$, for the study. We consider three fusions as the number of component networks is 8 and not very large so that evaluating all the component networks is computationally cheap.

The results are given in Table 2, and the depth of each component network is depicted in Table 1. It is observed that (i) the performance of the component network in the

[4]It is possible that the performance of the conventional ensemble with larger ensemble size is poorer, but generally it is better.

[5]In ensemble learning, the performance is related to the capability of each component classifier, while the performance depends more on the complementary of the base classifiers, i.e., the base classifiers make different kinds of errors.

Table 1. Performance variation when changing the component network capabilities. We vary the capabilities through varying the depths of deep base networks forming DFNs and accordingly the depths of the ensemble component networks. The depth of a component network in green (bue) means that the empirical performance of the corresponding network benefits (suffers) from the depth increase. The predicted relative capability of the DFNs are given in the column Pred. Rel. Cap.: $\uparrow$ ($\downarrow$, $\sim$) means the performance increase (decrease, undecided) compared with the DFN with a nearby smaller depth. It can be seen that *like the conventional ensemble with weight sharing, the experimental results of the DFNs are consistent to the predicted analysis*. Pred. Rel. Cap. = predicted relative capability. t. ensemble = tandem ensemble. c. ensemble = conventional ensemble of weight shared networks.

depth	LLL	RLL	LRL	LLR	RRL	RLR	LRR	RRR	Pred. Rel. Cap.	DFN (t. ensemble)	c. ensemble
14	5	8	8	8	11	11	11	14	–	7.99 –	8.00 –
20	5	10	10	10	15	15	15	20	$\uparrow$	7.20 $\uparrow$	7.07 $\uparrow$
26	5	12	12	12	19	19	19	26	$\uparrow$	7.06 $\uparrow$	6.92 $\uparrow$
38	5	16	16	16	27	27	27	38	$\sim$	8.09 $\downarrow$	6.96 $\downarrow$
50	5	20	20	20	35	35	35	50	$\sim$	8.79 $\downarrow$	7.03 $\downarrow$
56	5	22	22	22	39	39	39	56	$\downarrow$	8.85 $\downarrow$	7.36 $\downarrow$
62	5	24	24	24	43	43	43	62	$\downarrow$	9.04 $\downarrow$	7.79 $\downarrow$
74	5	28	28	28	51	51	51	74	$\downarrow$	9.41 $\downarrow$	8.14 $\downarrow$
98	5	36	36	36	67	67	67	98	$\downarrow$	9.95 $\downarrow$	8.80 $\downarrow$
110	5	40	40	40	75	75	75	110	$\downarrow$	13.2 $\downarrow$	10.1 $\downarrow$

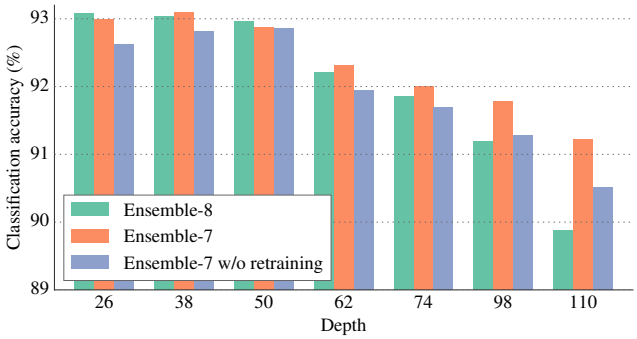


Figure 5. Comparing the performances of the ensembles with and without the deepest component network, *Ensemble-8* and *Ensemble-7*, and the ensemble without the deepest component network without retraining *Ensemble-7 w/o retraining* (the model parameters are the same to those of the large ensemble *Ensemble-8*). It can be seen that (i) the performances of *Ensemble-8* and *Ensemble-7* are very close, (ii) when the depth is larger than 50, *Ensemble-7* performs better than *Ensemble-8*, and (iii) *Ensemble-7* is superior to *Ensemble-7 w/o retraining*. These indicate that *the deepest network does not make the most significant contribution to the ensemble performance and it harms the training of other relatively shallow networks due to weight sharing*.

ensemble, whose depth is not very large, is deteriorated and (ii) the performances of other deeper networks, whose training is difficult without weight sharing, are boosted. In other words, *training the relatively shallow networks, due to sharing weights with the deep networks that are hard to train, is hurt while training the deep networks benefits from the shallow networks due to weight sharing*.

Deep Component Network. It is easy to show that a DFN formed from a deeper base network, with skipping a fixed number of layers (e.g., 2 suggested in ResNets [8]), leads to more fusions and thus a *larger ensemble size*. As a result, the performance becomes better, which is empirically validated by the result shown in Figure 4: the classification performance of the ResNet composed from a deeper base network is better.

We study how the deepest network influences the overall performance, by taking the DFN N_{f3} with three fusions as an example. We compare the performances of the large ensemble with the deepest component network, and the small ensemble without the deepest network, who are denoted by *Ensemble*-8 and *Ensemble*-7, respectively. To study how the deepest network affects the training of the ensemble (because of weight sharing), we also report the performance of the small ensemble without retraining it, i.e., the parameters are learnt from the large ensemble *Ensemble*-8, whose result is denoted by *Ensemble*-7 *w/o retraining*.

The results are given in Figure 5. On the one hand, one can see that the performances of the retrained small ensemble *Ensemble*-7 (the orange bar), where the deepest network is not included, are better than the unretrained small ensemble *Ensemble*-7 *w/o retraining* (the blue bar). It is another evidence that *the deepest network harms the training of other component networks in the ensemble because of weight sharing*. On the other hand, the retrained small ensemble *Ensemble*-7 performs closer to the large ensemble *Ensemble*-8, and even superior over the large ensemble *Ensemble*-8 in the cases of depths 98 and 110, because the deepest component network in the large ensemble harms the training of other shallow networks. This suggests that *the deepest network, though its performance is boosted with the help of other relatively shallow networks in the ensemble because of weight sharing, does not contribute most significantly to the overall performance*.

Furthermore, we study how each component network in the ensemble affects the performance. To this end, we report the performance by removing each component out from *Ensemble*-8 when testing. The results are shown in Table 3. The observations include: *the performances of the ensemble mainly depend on the networks of medium depths as*

Table 2. Illustrating how the performances of the deepest component network and other component networks are influenced by sharing weights across networks. It can be seen that the performances of the four component networks (LLL, RLL, LRL, LLR), which are relatively shallow, with weight sharing, are inferior to those without weight sharing, while the performances of the other four component networks (RRL, RLR, LRR, RRR), which are relatively deep, with weight sharing, are superior to those without weight sharing. It shows that *training the relatively shallow networks, due to sharing weights with the deep networks that are hard to train, is hurt, while training the deep networks benefits from the shallow networks due to weight sharing*. The overall performance of the ensemble without weight sharing is better, which is reasonable as without weight sharing, the correlation between ensemble component networks is lower. ↑ (↓) means the classification accuracy improvement (decrease) of the ensemble with weight sharing compared to the ensemble without weight sharing. "fail" means the classification error is higher than 50%.

Depth	Weight sharing	LLL	RLL	LRL	LLR	RRL	RLR	LRR	RRR	Overall
26	W/O	18.83	14.32	10.48	8.89	9.51	8.14	7.93	7.63	5.82
	W/	19.19 ↓	15.44 ↓	10.56 ↓	9.89 ↓	9.23 ↑	8.03 ↑	7.84 ↑	7.45 ↑	6.92 ↓
38	W/O	18.83	13.55	9.72	9.15	10.03	8.96	8.71	8.92	5.95
	W/	19.38 ↓	15.64 ↓	10.25 ↓	9.58 ↓	9.70 ↑	8.52	8.42 ↑	8.17 ↑	6.96 ↓
50	W/O	18.83	13.22	9.78	9.36	10.32	9.31	11.11	11.54	6.60
	W/	19.09 ↓	15.98 ↓	10.21 ↓	10.04 ↓	9.75 ↑	8.69 ↑	8.90 ↑	8.31 ↑	7.03 ↓
62	W/O	18.83	13.73	10.05	9.44	11.15	10.08	10.81	11.20	6.49
	W/	19.82 ↓	17.78 ↓	10.79 ↓	10.27 ↓	10.56 ↑	9.66 ↑	8.97 ↑	9.03 ↑	7.79 ↓
74	W/O	18.83	13.54	10.59	9.42	10.65	10.47	12.32	12.84	6.96
	W/	21.31 ↓	18.53 ↓	11.93 ↓	10.46 ↓	10.46 ↑	9.93 ↑	9.77 ↑	9.70 ↑	8.14 ↓
98	W/O	18.83	13.87	10.87	9.65	15.23	10.35	12.91	fail	7.21
	W/	21.26 ↓	19.18 ↓	11.61 ↓	9.77 ↓	11.65 ↑	9.31 ↑	11.78 ↑	11.68 ↑	8.80 ↓
110	W/O	18.83	14.28	11.07	9.56	14.72	10.89	17.94	fail	7.71
	W/	20.30 ↓	19.03 ↓	12.54 ↓	10.77 ↓	12.37 ↑	10.77 ↑	12.42 ↑	12.50 ↑	9.61 ↓

the performances are dropped significantly when the component networks of medium depths are removed; and the networks with small depths and large depths make less significant contribution.

5. Our Approach

The above study suggests that the performance of the deeply fused network depends on the capability of each component and the ensemble size and that the depth of the deepest component network is helpful to guarantee the ensemble size while itself does not contribute to the overall performance the most significantly. Inspired by the observations, we introduce an inception-like fusion manner and propose a novel merge-and-run fusion manner, to compose two networks[6]. Both are very simple and only uses the identity connection without introducing extra parameters.

Inception-like. The inception-like fusion unit is shown in Figure 6(a), which corresponds to two fusion units shown in Figure 6(c) in a ResNet. It is related to Inception [30] but different: the concatenation fusion is also used in [30] and the inception-like fusion only includes the summation fusion. Compared with the two fusion units, the inception-like contains only three paths, less than four paths in the two residual units, but the benefit is that the component networks are less deep.

Merge-and-run. The merge-and-run fusion unit is shown in Figure 6(b). It also corresponds to two fusion units shown in Figure 6(c) in a ResNet. It forms 4 paths, the same to two fusion units in a ResNet, and thus the ensemble size is the same to a ResNet in the same setup. The networks equipped with the merge-and-run fusion units are less deep and consists of more component networks of medium depths. In particular, compared with the inception-like fusion, the inputs to the two branches for each fusion unit are different, decreasing their correlation and thus improve the diversity of the tandem ensemble components. The merge-and-run fusion unit can be extended to use more than two branches, and we use the two branches version for comparisons.

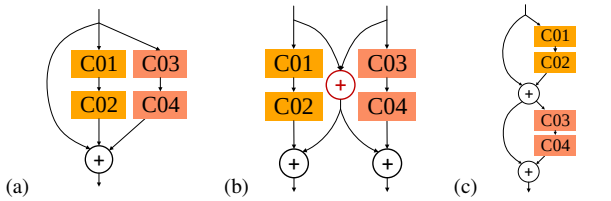


Figure 6. (a) Inception-like fusion; (b) Merge & run fusion; (c) two fusion units in ResNets.

Examples of two deeply-fused networks with the two fusion manners together with the ResNet with the same number of layers are given in Figure 7. The two new deep DFNs are less deep than the ResNet. Thus, the training is potentially easier (See convergence curves in Appendix B) and accordingly the performance is potentially better.

6. Experiments

We empirically show the superiority of the inception-like fusion and the merge-and-run fusion compared with

[6]The fusion schemes can be easily to be generalized to compose more networks which potentially could be better. Due to space limitation, this paper focuses on the case of combining two networks.

Table 3. The importance of the deepest component network and other component networks. The performance by removing each component network (denoted by W/O) from the ensemble is reported. It can be observed that *the deepest component network (RRR) contributes to the overall performance less significantly than the medium-deep networks (LLR, RLR, LRR) as the error when removing LLR, RLR, and LRR out is the largest except the case of depth* 26. The largest classification error for each depth is in bold.

Depth	W/O LLL	W/O RLL	W/O LRL	W/O LLR	W/O RRL	W/O RLR	W/O LRR	W/O RRR	Overall
26	6.78	6.81	7.00	6.92	6.98	7.19	7.21	**7.34**	6.92
38	6.85	6.90	6.85	**7.22**	6.85	7.15	7.02	7.18	6.96
50	7.06	7.08	6.89	7.34	7.00	**7.38**	7.15	7.14	7.03
62	7.83	7.72	7.61	7.90	7.62	7.95	**8.10**	8.05	7.79
74	7.87	7.85	8.04	8.13	8.36	**8.48**	8.43	8.37	8.14
98	8.92	8.90	8.63	9.64	8.74	**9.75**	8.65	8.72	8.80
110	9.70	9.65	9.39	**10.20**	9.36	10.14	9.46	9.48	9.61

ResNets. We demonstrate the effectiveness of deeply-fused nets with the merge-and-run fusion on several benchmark datasets and compare it with the state-of-the-arts.

6.1. Datasets

CIFAR-10 **and CIFAR-**100**.** The two datasets are both subsets [13] drawn from the 80-million tiny image database [34]. CIFAR-10 consists of 60000 32×32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. CIFAR-100 is like CIFAR-10, except that it has 100 classes each containing 600 images. We follow a standard data augmentation scheme widely used for this dataset [16, 8, 10]: we first zero-pad the images with 4 pixels on each side, and then randomly crop to produce 32×32 images, followed by horizontally mirroring half of the images. We preprocess the images by normalizing the images using the channel means and standard deviations.

SVHN. The SVHN (street view house numbers) dataset consists of digit images of size 32×32. There are $73,257$ images as the training set, $531,131$ images as a additional training set, and $26,032$ images as the testing set. Following the common practice [17, 16, 11], we select out 400 samples per class from the training set and 200 samples per class from the additional set, and use the remaining $598,388$ images for training.

6.2. Training Setup

We use the SGD algorithm with the Nesterov momentum to train all the models for 400 epochs on CIFAR-10/CIFAR-100 and 40 epochs on SVHN, both with a total mini-batch size 64 on two GPUs. The learning rate starts with 0.1 and is reduced by a factor 10 at the $1/2$, $3/4$ and $7/8$ fractions of the number of training epochs. Similar to [8, 10], the weight decay is 0.0001, the momentum is 0.9, and the weights are initialized as in [7]. We follow ResNets to design our networks: use three groups of fusion units with the number of filter channels being $16, 32, 64$, respectively, and use a Conv-BN-ReLU as a basic layer with the filter size of 3×3. In the merge-and-run fusion, merge ($+$ in Figure 6(b)) is an averaging operation. All the other $+$ operations in Figures 6 are between BN and ReLU. Our implementation is based on MXNet [2], and the code is available at https://github.com/zlmzju/fusenet.

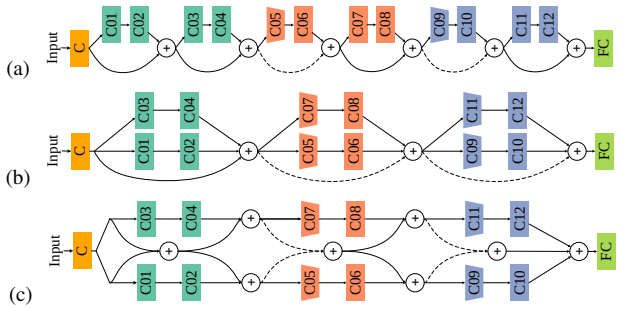


Figure 7. Network examples: (a) a ResNet; (b) a DFN with inception-like fusion; (c) a DFN with merge-and-run fusion. The trapezoid shape indicates that down-sampling (stride=2) occurs in the layer, and the dashed line denotes a projection shortcut as in [8].

6.3. Empirical Study

We compare the two DFNs with the inception-like fusion and the merge-and-run fusion, denoted by DFN-IL and DFN-MR, and the baseline ResNet algorithm. They have the same number of parameters/layers, and each fusion unit in a DFN-IL and a DFN-MR corresponds to two fusion units in a ResNet. Examples of the three nets are shown in Figure 7.

The comparison on CIFAR-10 is given in Table 4. One can see that compared with ResNets, DFN-MRs and DFN-ILs consistently perform better, and DFN-MRs perform the best. The superiority of DFN-ILs over ResNets stems from the overall higher capability of each ensemble component network: the component networks in a DFN-IL are also deep but relatively shallow and thus easy to train. The additional benefits of a DFN-MR include that its ensemble size is greater than that of a DFN-IL and that the component networks are less correlated. In particular, we can see that the superiority of DFNs to ResNets is more significant when the nets are deeper or with more layers, which implies that DFNs benefit more from the increasing number of layers.

The comparisons over CIFAR-100 and SVHN shown in

Table 4. Empirical comparison of our approaches, DFN-ILs and DFN-MRs, with ResNets, in terms of the average classification error from 5 runs and the standard deviation (mean $\pm$ std.). For d_1/d_2 in the column Depth, d_1 means the depth of the plain network and the ResNet and d_2 means the depth of the corresponding DFN-IL (DFN-MR). See Figure 7 illustrating the three nets. The lowest classification errors are in bold. "fail" means the classification error is higher than 50%.

Params.	Depth	CIFAR-10				CIFAR-100			SVHN		
		Plain	ResNet	DFN-IL	DFN-MR	ResNet	DFN-IL	DFN-MR	ResNet	DFN-IL	DFN-MR
0.4M	26/14	7.88 ± 0.19	6.62 ± 0.24	6.53 ± 0.12	$\mathbf{6.48 \pm 0.04}$	29.69 ± 0.15	29.75 ± 0.27	$\mathbf{29.62 \pm 0.08}$	$\mathbf{1.90 \pm 0.08}$	2.13 ± 0.09	2.00 ± 0.04
0.6M	38/20	8.78 ± 0.28	5.93 ± 0.17	5.83 ± 0.09	$\mathbf{5.79 \pm 0.13}$	27.90 ± 0.26	27.87 ± 0.22	$\mathbf{27.80 \pm 0.26}$	1.97 ± 0.09	1.96 ± 0.10	$\mathbf{1.87 \pm 0.09}$
0.8M	50/26	11.04 ± 0.54	5.60 ± 0.14	5.59 ± 0.17	$\mathbf{5.47 \pm 0.14}$	27.03 ± 0.66	26.88 ± 0.22	$\mathbf{26.76 \pm 0.16}$	1.93 ± 0.17	1.86 ± 0.14	$\mathbf{1.86 \pm 0.05}$
1.0M	62/32	11.92 ± 0.49	5.50 ± 0.09	5.45 ± 0.09	$\mathbf{5.10 \pm 0.08}$	26.44 ± 0.69	26.19 ± 0.41	$\mathbf{25.87 \pm 0.04}$	1.89 ± 0.03	1.86 ± 0.07	$\mathbf{1.81 \pm 0.04}$
1.2M	74/38	13.83 ± 0.86	5.35 ± 0.14	5.26 ± 0.20	$\mathbf{5.18 \pm 0.20}$	26.00 ± 0.48	25.98 ± 0.23	$\mathbf{25.41 \pm 0.19}$	1.90 ± 0.04	1.81 ± 0.11	$\mathbf{1.77 \pm 0.11}$
1.5M	98/50	fail	5.26 ± 0.09	5.05 ± 0.20	$\mathbf{4.99 \pm 0.13}$	25.44 ± 0.20	24.76 ± 0.33	$\mathbf{24.73 \pm 0.40}$	1.91 ± 0.03	1.84 ± 0.06	$\mathbf{1.84 \pm 0.15}$
1.7M	110/56	fail	5.24 ± 0.23	5.02 ± 0.17	$\mathbf{4.96 \pm 0.06}$	24.56 ± 0.15	24.56 ± 0.69	$\mathbf{24.41 \pm 0.09}$	2.00 ± 0.12	1.85 ± 0.09	$\mathbf{1.68 \pm 0.02}$
3.1M	194/98	fail	5.47 ± 0.46	4.97 ± 0.03	$\mathbf{4.84 \pm 0.11}$	24.41 ± 0.10	24.06 ± 0.68	$\mathbf{23.98 \pm 0.15}$	1.85 ± 0.03	1.75 ± 0.06	$\mathbf{1.70 \pm 0.03}$

Table 5. Classification error comparison with state-of-the-arts. DFN-MR1 is our approach with merge-and-run fusion, and DFN-MR2 and DFN-MR3 are the wide version of our approach ($4\times$ wider). DO = dropout, DP = droppath, PA = pre-activation, SD = stochastic depth. C10 = CIFAR-10, C100 = CIFAR-100.

	Depth	Params	C10	C100	SVHN
NIN [17]	-	-	8.81	-	2.35
All-CNN [26]	-	-	7.25	33.71	-
FitNet [22]	-	-	8.39	35.04	2.42
DSN [16]	-	-	7.97	34.57	1.92
Swapout [24]	20	1.1M	6.85	25.86	-
	32	7.4M	4.76	22.72	-
Highway [28]	-	-	7.72	32.39	-
DFN [36]	50	3.7M	6.40	27.61	-
	50	3.9M	6.24	27.52	-
FractalNet [15]	21	38.6M	5.22	23.30	2.01
W/ DO & DP	21	38.6M	4.60	23.73	1.87
ResNet [8]	110	1.7M	6.61	-	-
ResNet [11]	110	1.7M	6.41	27.22	2.01
ResNet (PA) [9]	164	1.7M	5.46	24.33	-
	1001	10.2M	4.62	22.71	-
ResNet W/ SD [11]	110	1.7M	5.23	24.58	1.75
	1202	10.2M	4.91	-	-
Wide ResNet [38]	16	11.0M	4.81	22.07	-
	28	36.5M	4.17	20.50	-
W/ DO	16	2.7M	-	-	1.64
RiR [33]	18	10.3M	5.01	22.90	-
Multi ResNet [1]	200	10.2M	4.35	20.42	-
	398	20.4M	3.92	-	-
DenseNet [10]	100	27.2M	3.74	19.25	1.59
DFN-MR1 (ours)	56	1.7M	4.94	24.46	1.66
DFN-MR2 (ours)	32	14.9M	3.94	**19.25**	**1.51**
DFN-MR3 (ours)	50	24.8M	**3.57**	**19.00**	**1.55**

Table 4 are consistent to the comparisons over CIFAR-10: with more layers the superiority of our approach is more significant. One exception is that on SVHN the ResNet of depth 26 performs better than the DFNs. The reason might be that the DNF of depth 14 is not very deep and that too many shallow component networks in the corresponding ensemble on the particular SVNH dataset lower down the performance.

6.4. Comparison with State-of-the-Arts

The comparison is reported in Table 5. We report the results of DFN-MRs since it is superior to DFN-ILs with only a run due to the expensive computation cost for training the wide versions. We mark the results that outperform existing state-of-the-arts in bold and the best results in blue.

One can see that the wide version DFN-MR3 ($4\times$ wider, #channels are $64, 128, 256$, the depth is 50) outperforms existing state-of-the-art results and achieves the best results on CIFAR-10 and CIFAR-100. Compared with the second best approach DenseNet that includes more parameters ($27.2M$), our net includes only $24.8M$ parameters.

The wide and less deep version DFN-MR2 (depth $= 32$) is also very competitive: outperform all existing state-of-the-art results on SVHN, perform the second best together with DenseNet on CIFAR-100, and inferior only to Multi ResNet and DenseNet (both of which include much more parameters) on CIFAR-10. It includes only $14.9M$ parameters, almost half of the parameters ($27.2M$) of the competitive DenseNet.

Compared with the FractalNet, DFN-MR2 and DFN-MR3 are deeper ($32, 50$, more difficult to train) and include fewer parameters. They achieve superior performances, which suggests that our approaches suffer less from the training difficulty due to the great depth and exploit the parameters and the depth more effectively. Different from the FractalNet composed of fractal fusion units, which can be regarded as a kind of deep fusion, the DFN-MR adopts the merge-and-run fusion, which is more beneficial for improving the overall ensemble performance.

6.5. Results on ImageNet

We compare our proposed network, DFN-MR (deeply-fused nets with merge-and-run fusion), against ResNet on the ImageNet 2012 classification dataset [4], which consists of 1000 classes of images. The models are trained on the 1.28 million training images, and evaluated on the $50,000$ validation images.

Network architecture. In this experiment, we compare the results of the ResNet with 101 layers and will report results with other numbers of layers in the future. The ResNet-101 [8] (44.5M) is equipped with 4 groups of bottleneck blocks, and the numbers of blocks in the four groups are $(3, 4, 23, 3)$, respectively. We form our DFN-MR network by replacing the ResNet fusion blocks with our merge-and-

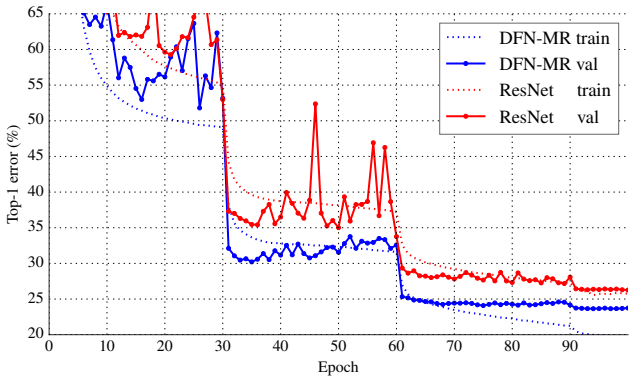


Figure 8. Training error and validation error curves of ResNet-101 (44.5M) and DFN-MR (43.3M) with the same optimization setting on ImageNet. We report the (top-1, top-5) results for both training error and single-crop validation error. It can be observed that our approach performs better for both training errors and validation errors.

run fusion blocks and setting the numbers of blocks in the four groups to $(2, 2, 8, 2)$, resulting in a DFN-MR network (43.3M). The two networks are illustrated in Figure 9.

Optimization. We follow [8] and use SGD to train the two models using the same hyperparameters (weight decay $=$ 0.0001, and momentum $=$ 0.9) with [8]. The mini-batch size is 256, and we use 8 GPUs (32 samples per GPU). We adopt the same data augmentation as in [8]. We train the models for 100 epochs, and start from a learning rate of 0.1, and then divide it by 10 every 30 epochs which are the same as the learning rate changing in [8, 6]. We evaluate on the single 224×224 center crop from an image whose shorter side is 256.

Results. The training error and validation error curves of ResNet-101 and DFN-MR are given in Figure 8. It can be observed that our approach performs better for both training errors and validation errors. For example, the top-1 validation error of our approach is lower about 5% than that of the ResNet from the 30th epoch to the 55th epoch. We have similar observations from CIFAR and SVHN, as shown in Figure 12.

Table 6 shows the results of our approach, our MXNet implementation of ResNet-101, and the result of ResNet-101 from [8]. We can see that our approach performs the best in terms of top-5 validation error: our approach gets 1.7 gain, compared with the results of ResNet-101 from our implementation, and 0.3 gain compared with the result from [8].

We notice that the results of our implemented ResNet on MXNet, and the results from [8] are different. We also want to point out that the settings are the *same* with [8]. We think that the difference might be from the MXNet platform, or there might be some other untested issues as pointed in https://github.com/KaimingHe/deep-residual-networks.

Table 6. The validation (single 224×224 center crop) and training performances (%) of ResNet-101 (44.5M) and DFN-MR (43.3M) shown in Figure 9 on ImageNet.

	ResNet-101 [8]	ResNet-101	DFN-MR
#parameters	44.5M		43.3M
Top-1 validation error	23.60	26.41	23.66
Top-5 validation error	7.10	8.50	**6.81**
Top-1 training error	17.00	25.75	19.72
Top-5 training error	-	8.12	6.59

7. Conclusion

In this paper, we study the connection of deep fusion to ensembling from architecture resemblance and performance variation resemblance. We empirically observe that the deepest component in the ensemble does not play the most significant role on the overall performance improvement, while the depth increase is beneficial to enlarge the ensemble size. Finally, we propose a novel network architecture, deeply-fused nets with merge-and-run fusion, which is helpful to guarantee ensemble size and ensemble component network capability. Empirical results show that our approach achieves the state-of-the-art results.

A. On the Connection to Ensembling

A.1. Results on ImageNet

In the main paper, we show that the performance of deeply-fused nets is related to the ensemble size and the capability of ensemble components on CIFAR-10. Here, we provide some empirical results on ImageNet. We compare ResNet-50 and its variants, DFN1, DFN2, shown in Figure 10.

We train the three networks using the same setting. We use SGD with a mini-batch size of 256 on 4 GPUs (64 per GPU). The weight decay is 0.0001 and the momentum is 0.9. We start from a learning rate of 0.1, and divide it by 2 every 10 epochs [7].

Analysis. We compare the three networks in terms of ensemble sizes and component network capabilities.

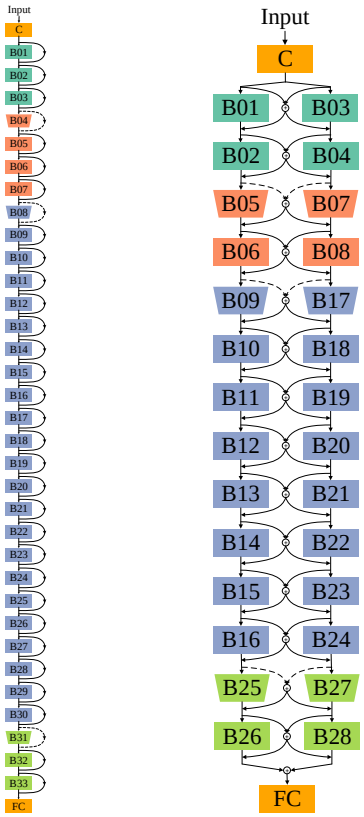
On the one hand, we have the following property about the ensemble sizes

$$\begin{aligned} & \text{Ensemble Size}(\text{ResNet-50}) \\ = \; & \text{Ensemble Size}(\text{DFN1}) \\ > \; & \text{Ensemble Size}(\text{DFN2}). \end{aligned} \tag{1}$$

On the other hand, we analyze the component network capabilities by comparing the distributions of the depths of ensemble component networks, shown in Figure 11. We have the following observations:

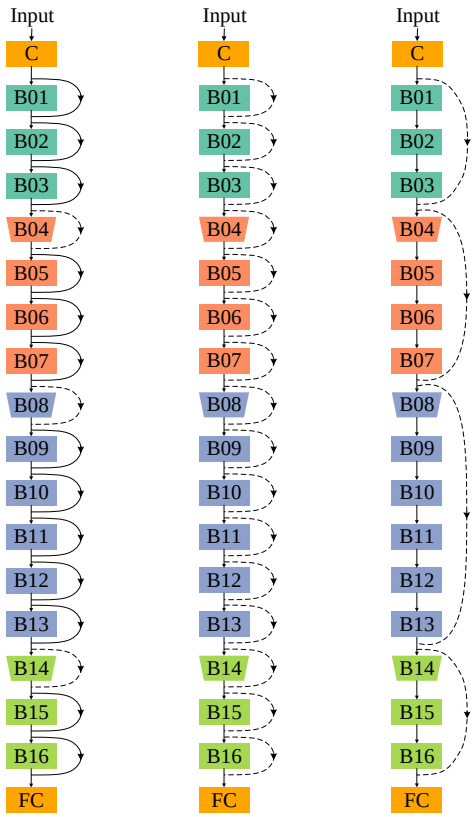
[7] The learning rate decreasing scheme for this study is slightly different from the scheme in ResNet [8] and *fb.resnet.torch* [6]. We note that all the comparisons are done in the same training settings for the three networks.

We fail to reproduce the results reported in [8] by using the provided settings, which is also observed in the ResNet-101 in Section 6.5.



(a) ResNet-101 (b) DFN-MR

Figure 9. The architecture of the networks used for ImageNet classification. (a) ResNet-101 with 4 groups of bottleneck blocks. (b) Our DFN-MR network, which corresponds to a ResNet-101 network with a similar (but smaller) number of parameters.
Each block, B01, B02, . . . , is a 3-layer bottleneck block [8]. The dashed line indicates the projection shortcut with a 1×1 convolutional layer. The pooling layers are omitted for simplicity.



(a) ResNet-50 (b) DFN1 (c) DFN2

Figure 10. The architecture of the 50-layer networks used for ImageNet classification. Each block, B01, B02, . . . , is a 3-layer bottleneck block [8]. The dashed line indicates the projection shortcut with a 1×1 convolutional layer. The pooling layers after the first convolutional layer C and before the fully-connected layer FC are omitted for simplicity

- The depths of the ensemble component networks from ResNet-50 are more diverse and more uniformly distributed: training the very deep component networks (deeper than 20) benefit from many shallow networks (See Section 4 and Tables 2 and 3 in the main paper).

- Compared to ResNet-50, the ensemble component networks from DFN1 are much deeper, and there are a few shallow networks that are still deeper than the shallow networks from ResNet-50, and thus training the deeper component networks is not as easy as that in ResNet-50.

- The ensemble component networks from DFN2 all belong to the ensemble component networks from ResNet-50, but the ensemble size is much smaller.

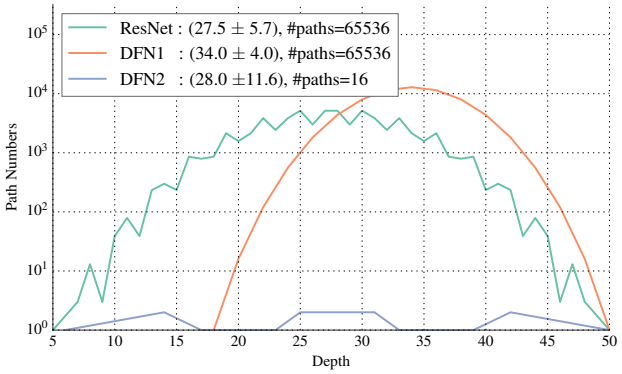


Figure 11. The distributions of the depths of ensemble component networks. Different networks: (avg depth ± std).

Taken together, among ResNet-50, DFN1 and DFN2, DFN2 probably performs the poorest because the ensemble size is the smallest, and DFN1 probably performs inferiorly to ResNet-50 because of the lower capability of component networks. The analysis is validated by the empirical performance, presented in Table 7.

Table 7. The validation (single 224×224 center crop) and training performances (%) of the DNFs shown in Figure 10 on ImageNet.

	ResNet-50 [8]	ResNet-50	DFN1	DFN2
Top-1 validation error	24.70	24.71	24.99	27.12
Top-5 validation error	7.80	7.40	7.79	8.96
Top-1 training error	20.10	18.79	17.54	23.72
Top-5 training error	-	5.73	5.25	8.06

B. Convergence Process

Figure 12 shows the converge curves of a ResNet and a DFN-MR with the same number of layers/parameters. It can be seen that the training loss of DFN-MR decreases more rapidly and the test error also decreases more rapidly.

References

[1] M. Abdi and S. Nahavandi. Multi-residual networks. *CoRR*, abs/1609.05672, 2016. 2, 8

[2] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, abs/1512.01274, 2015. 7

[3] D. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *CoRR*, abs/1511.07289, 2015. 2

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 8

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 1

[6] S. Gross and M. Wilber. Training and investigating residual nets. `https://github.com/facebook/fb.resnet.torch`, 2016. 9

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 2, 7

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 2, 4, 5, 7, 8, 9, 10, 11

[9] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 2, 8

[10] G. Huang, Z. Liu, and K. Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. 2, 7, 8

[11] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger. Deep networks with stochastic depth. *CoRR*, abs/1603.09382, 2016. 2, 7, 8

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 2

[13] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 7

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[15] G. Larsson, M. Maire, and G. Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *CoRR*, abs/1605.07648, 2016. 1, 2, 8

[16] C. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, 2015. 7, 8

[17] M. Lin, Q. Chen, and S. Yan. Network in network. *CoRR*, abs/1312.4400, 2013. 2, 7, 8

[18] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[19] I. Loshchilov and F. Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016. 2

[20] D. Mishkin and J. Matas. All you need is a good init. *CoRR*, abs/1511.06422, 2015. 2

[21] M. Pezeshki, L. Fan, P. Brakel, A. C. Courville, and Y. Bengio. Deconstructing the ladder network architecture. In *ICML*, pages 2368–2376, 2016. 2

[22] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *CoRR*, abs/1412.6550, 2014. 8

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 2

[24] S. Singh, D. Hoiem, and D. A. Forsyth. Swapout: Learning an ensemble of deep architectures. *CoRR*, abs/1605.06465, 2016. 8

[25] L. N. Smith and N. Topin. Deep convolutional neural network design patterns. *CoRR*, abs/1611.00847, 2016. 2

[26] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014. 8

[27] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 2

[28] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. In *NIPS*, pages 2377–2385, 2015. 1, 2, 8

[29] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 1139–1147, 2013. 2

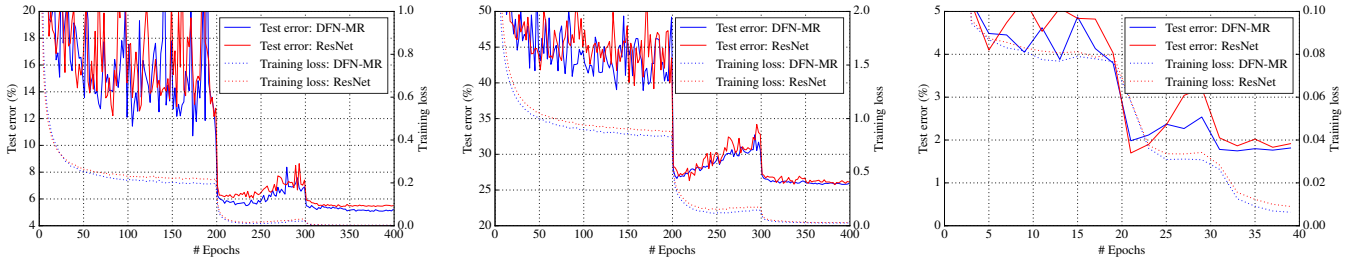
[30] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016. 2, 6

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 1, 2

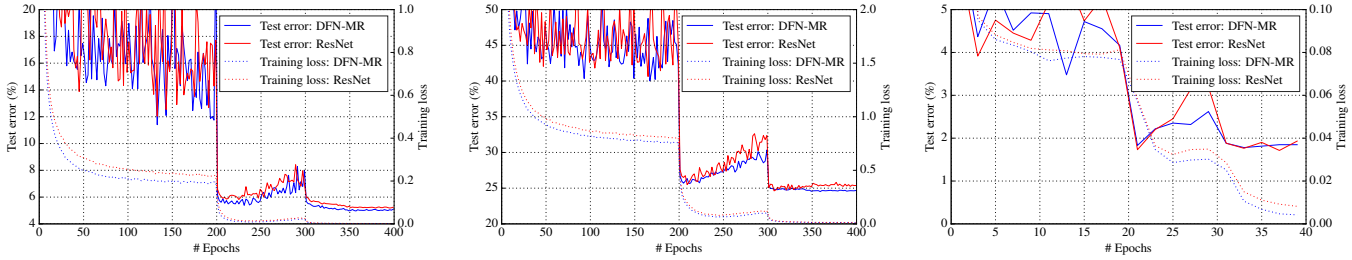
[32] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 2

[33] S. Targ, D. Almeida, and K. Lyman. Resnet in resnet: Generalizing residual architectures. *CoRR*, abs/1603.08029, 2016. 2, 8

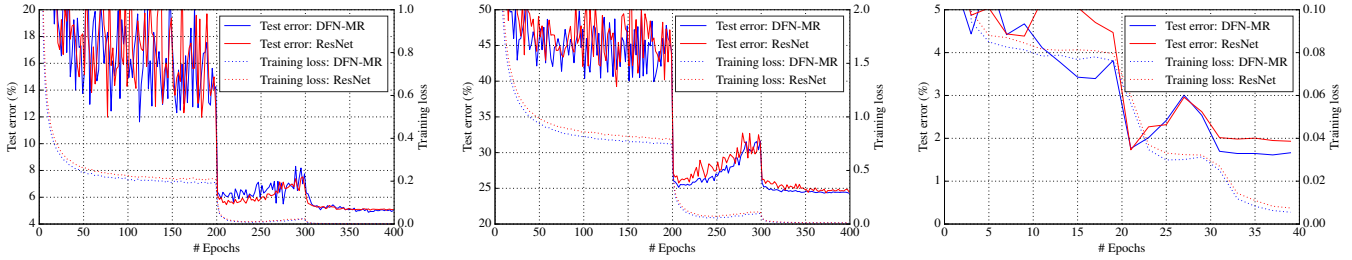
[34] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and



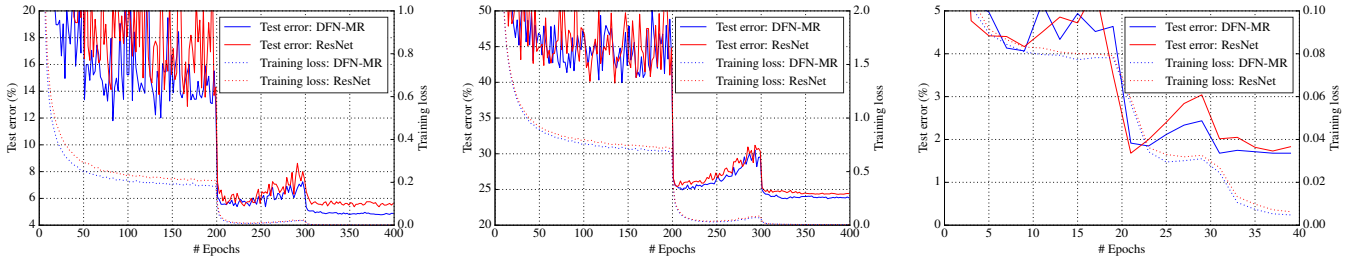
(a) CIFAR-10 (62/32) (b) CIFAR-100 (62/32) (c) SVHN (62/32)



(d) CIFAR-10 (98/50) (e) CIFAR-100 (98/50) (f) SVHN (98/50)



(g) CIFAR-10 (110/56) (h) CIFAR-100 (110/56) (i) SVHN (110/56)



(j) CIFAR-10 (194/98) (k) CIFAR-100 (194/98) (l) SVHN (194/98)

Figure 12. Comparing the optimization of the ResNet and the DFN-MR with the same number of layers/parameters for various depths. d_1/d_2 = (the depth of a ResNet) / (the depth of the corresponding of a DFN-MR), and $d_2 = \frac{d_1}{2} + 1$. The vertical axis corresponds to training losses(testing errors), and the horizontal axis corresponds to #epochs.

scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(11):1958–1970, 2008. 7

[35] A. Veit, M. J. Wilber, and S. J. Belongie. Residual networks are exponential ensembles of relatively shallow networks. *CoRR*, abs/1605.06431, 2016. 1, 2

[36] J. Wang, Z. Wei, T. Zhang, and W. Zeng. Deeply-fused nets. *CoRR*, abs/1605.07716, 2016. 1, 2, 8

[37] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, pages 1395–1403, 2015. 1

[38] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016. 2, 4, 8

[39] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu. Residual networks of residual networks: Multilevel residual networks. *CoRR*, abs/1608.02908, 2016. 2