Deep Learning 101

4주차

12기 이두형
12기 임효진

Curriculum

1주차 : 딥러닝 소개 및 기초 (XOR문제, 퍼셉트론, 활성화 함수 등)

2주차 : Multi-layer Neural Network (Loss Function, Gradient Descending, Backpropagation, MNIST practice, Optimization)

3주차 : CNN 소개 및 기초 (Convolution, Padding, Stride, Pooling등 기초 개념 소개)

4주차 : CNN 실습 (CIFAR-10)

5주차 : RNN, LSTM, GRU

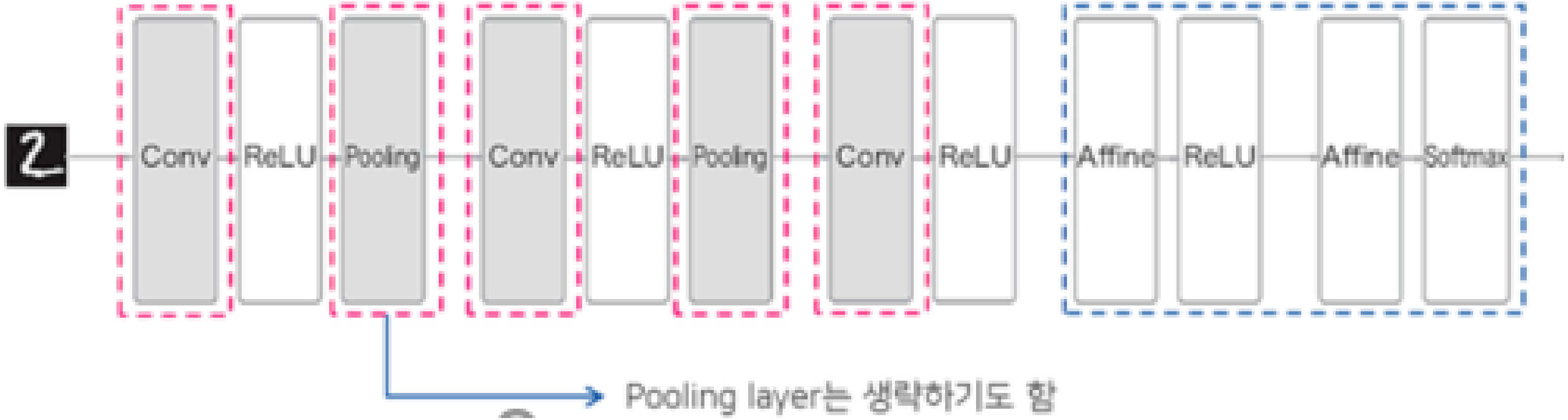
6주차 : seq2seq, 실습 (세션 후 조별 과제 부여)

7주차 : 조별 과제 발표

Review

Review

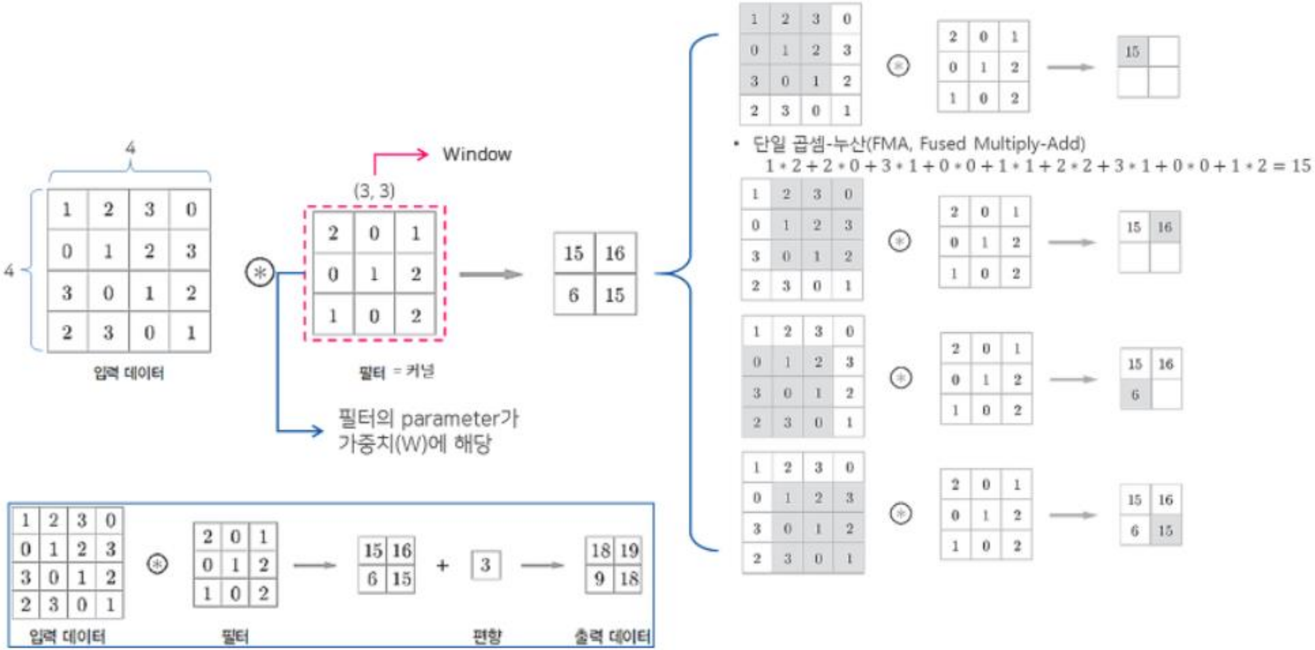
- CNN



Pooling layer는 생략하기도 함

Review

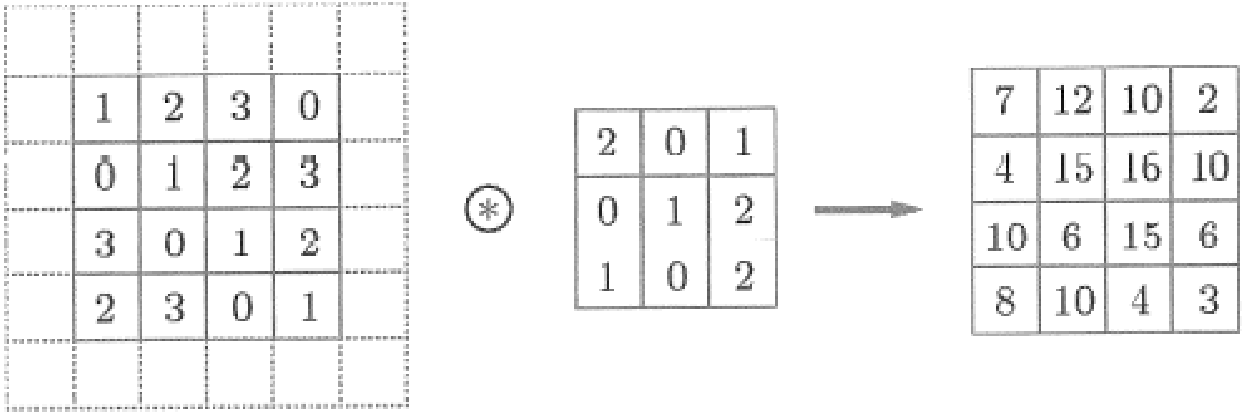
- Convolution



$$\begin{bmatrix}1&2&3&0\\0&1&2&3\\3&0&1&2\\2&3&0&1\end{bmatrix} \circledast \begin{bmatrix}2&0&1\\0&1&2\\1&0&2\end{bmatrix} \rightarrow \begin{bmatrix}15&16\\6&15\end{bmatrix} + \boxed{3} \rightarrow \begin{bmatrix}18&19\\9&18\end{bmatrix}$$

Review

- Padding



1	2	3	0
0	1	2	3
3	0	1	2
2	3	0	1

(4, 4)

입력 데이터(패딩 : 1)

2	0	1
0	1	2
1	0	2

(3, 3)

필터

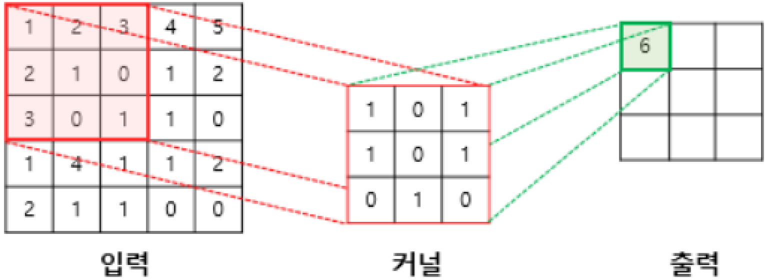
7	12	10	2
4	15	16	10
10	6	15	6
8	10	4	3

(4, 4)

출력 데이터

Review

- Stride



1	2	3	4	5
2	1	0	1	2
3	0	1	1	0
1	4	1	1	2
2	1	1	0	0

입력

1	0	1
1	0	1
0	1	0

커널

6		

출력

Review

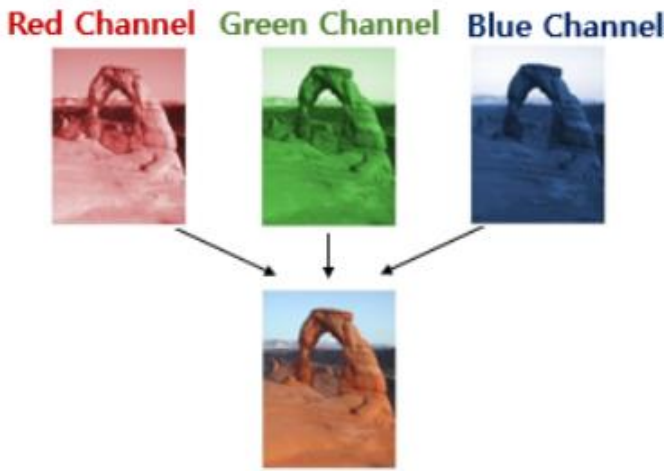
- Feature map size

$$OH = \frac{H + 2P - FH}{S} + 1$$

$$OW = \frac{W + 2P - FW}{S} + 1$$

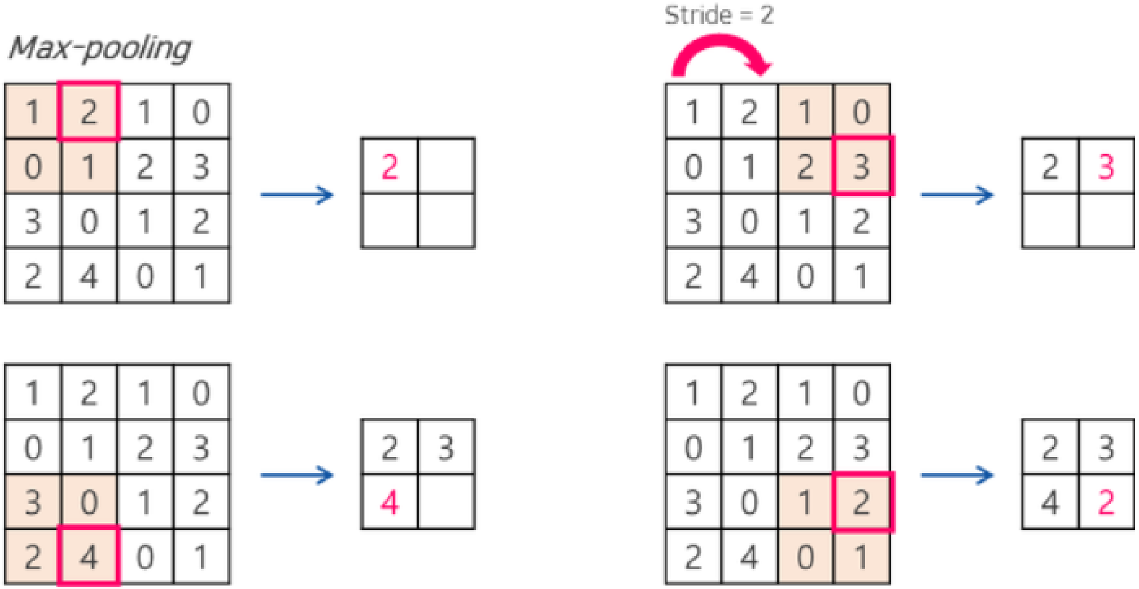
Review

- Channel



Review

- Pooling



1	2	1	0
0	1	2	3
3	0	1	2
2	4	0	1

2	

2	3

2	3
4	

2	3
4	2

[그림 7] Max-pooling 예제

Review

- LeNet & AlexNet

그림 7-28 AlexNet의 구성[21]

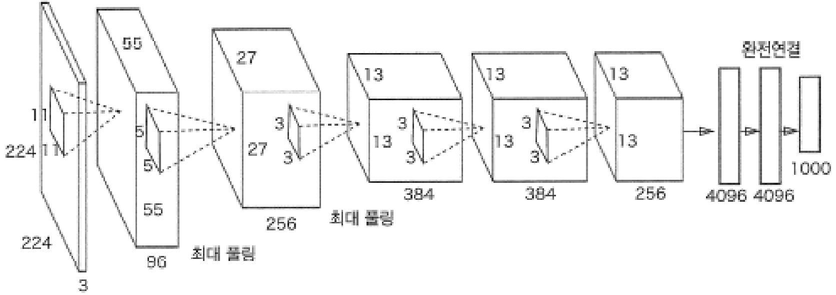
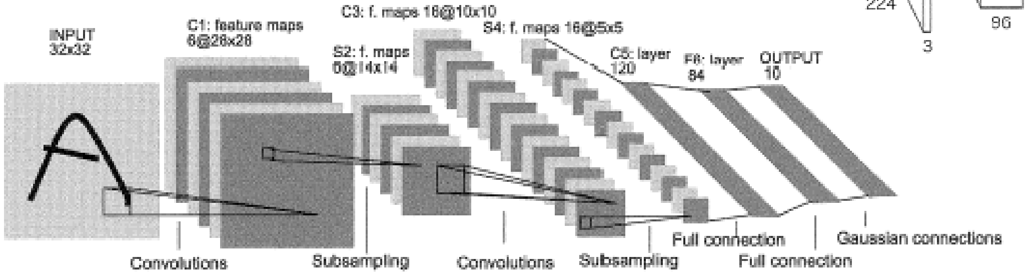


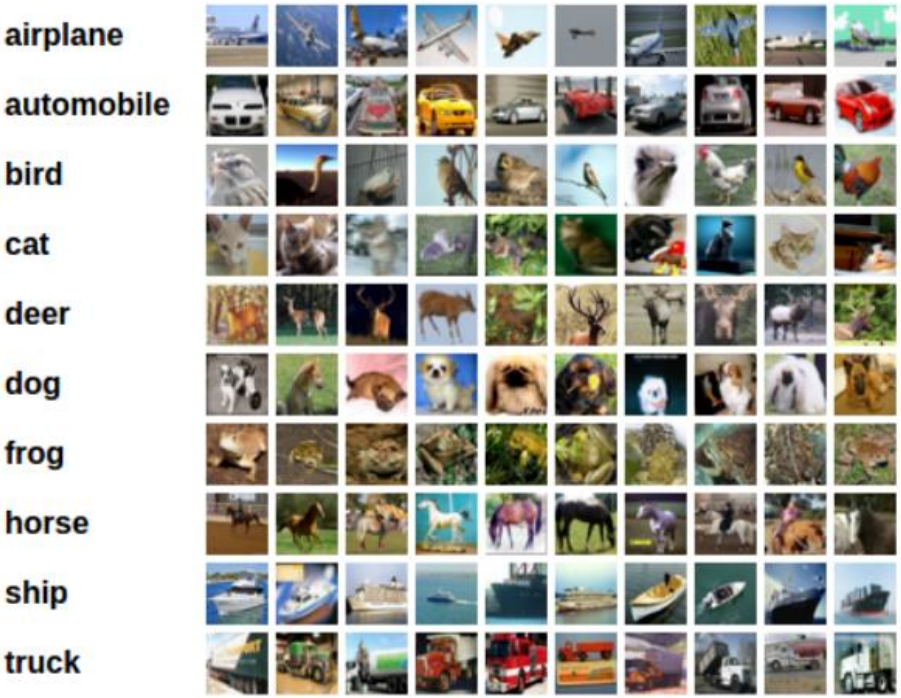
그림 7-27 LeNet의 구성[20]



Week4

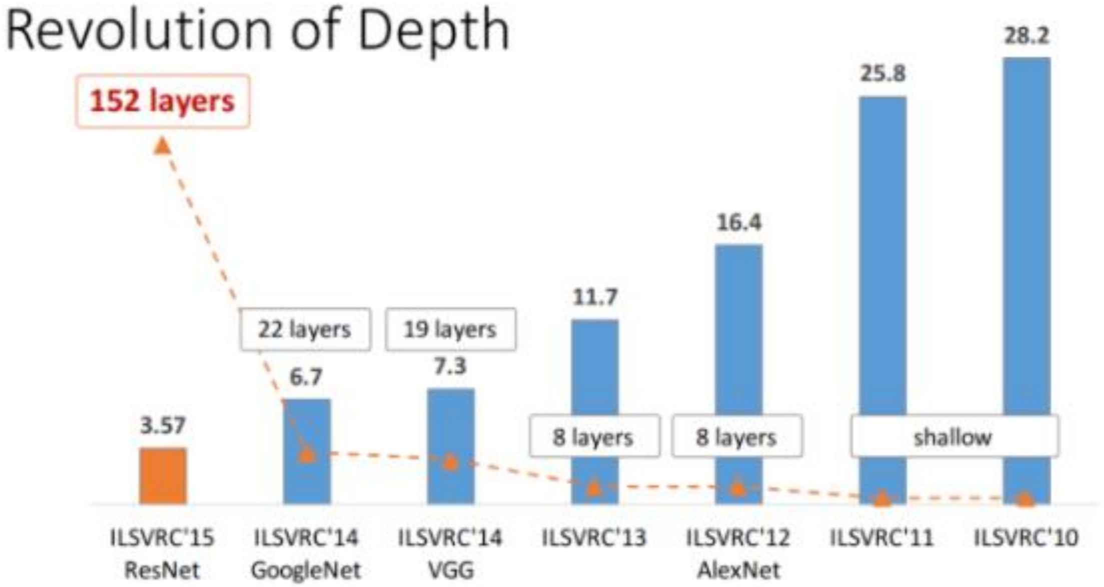
CIFAR-10

- CIFAR-10



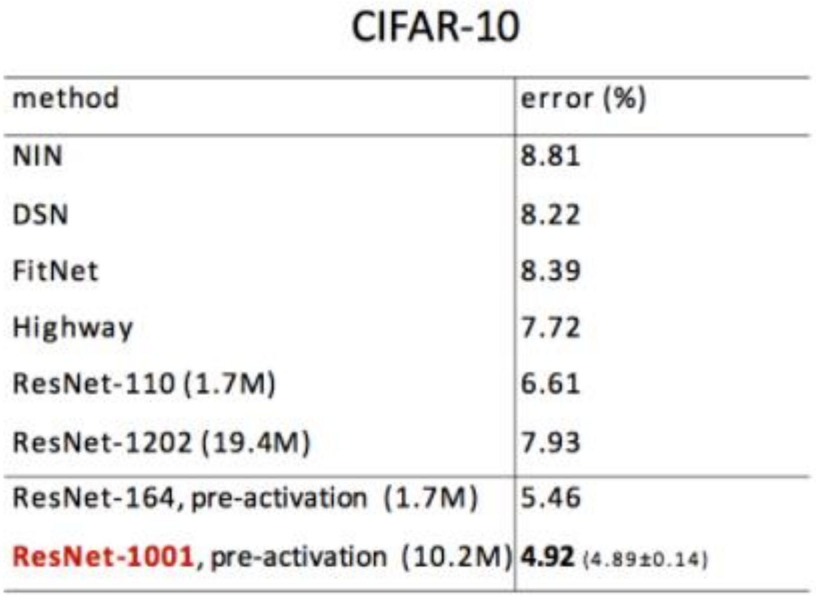
CIFAR-10

- CIFAR-10



CIFAR-10

- CIFAR-10

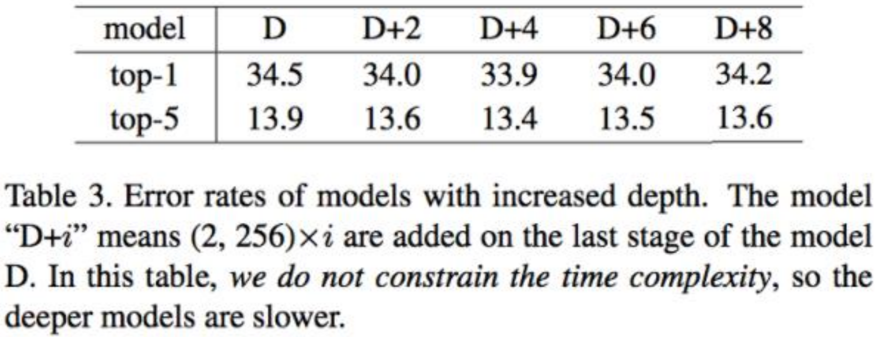


CIFAR-10

method	error (%)
NIN	8.81
DSN	8.22
FitNet	8.39
Highway	7.72
ResNet-110 (1.7M)	6.61
ResNet-1202 (19.4M)	7.93
ResNet-164, pre-activation (1.7M)	5.46
ResNet-1001, pre-activation (10.2M)	**4.92** (4.89±0.14)

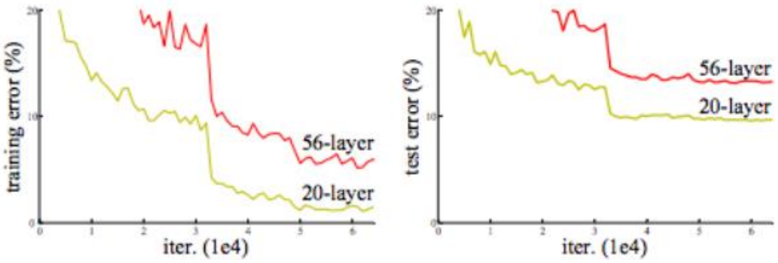
ResNet

- Degradation



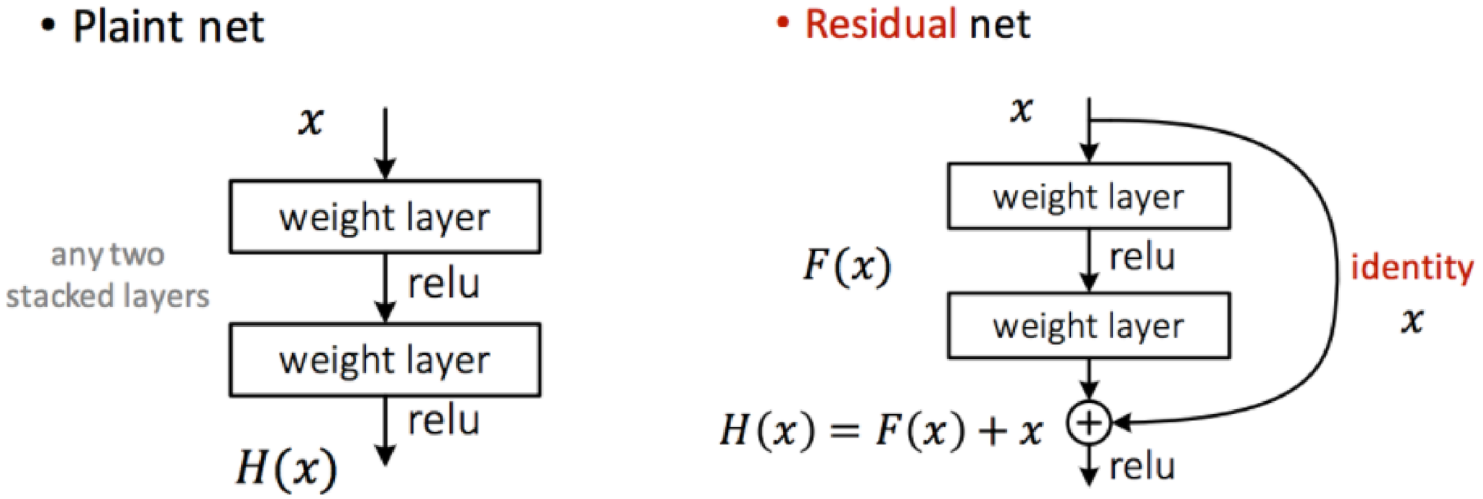
model	D	D+2	D+4	D+6	D+8
top-1	34.5	34.0	33.9	34.0	34.2
top-5	13.9	13.6	13.4	13.5	13.6

Table 3. Error rates of models with increased depth. The model "D+i" means (2, 256)×i are added on the last stage of the model D. In this table, *we do not constrain the time complexity*, so the deeper models are slower.



ResNet

- ResNet

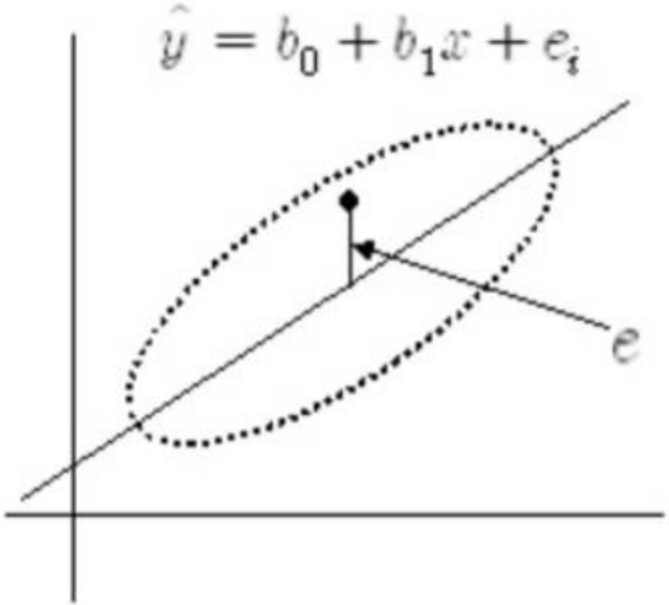


- Plaint net

- Residual net

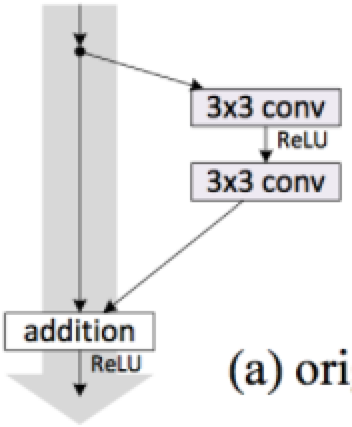
ResNet

- Residual

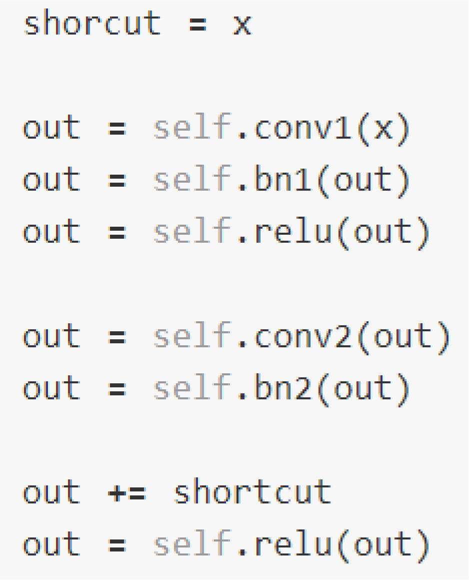


ResNet

- Residual Block



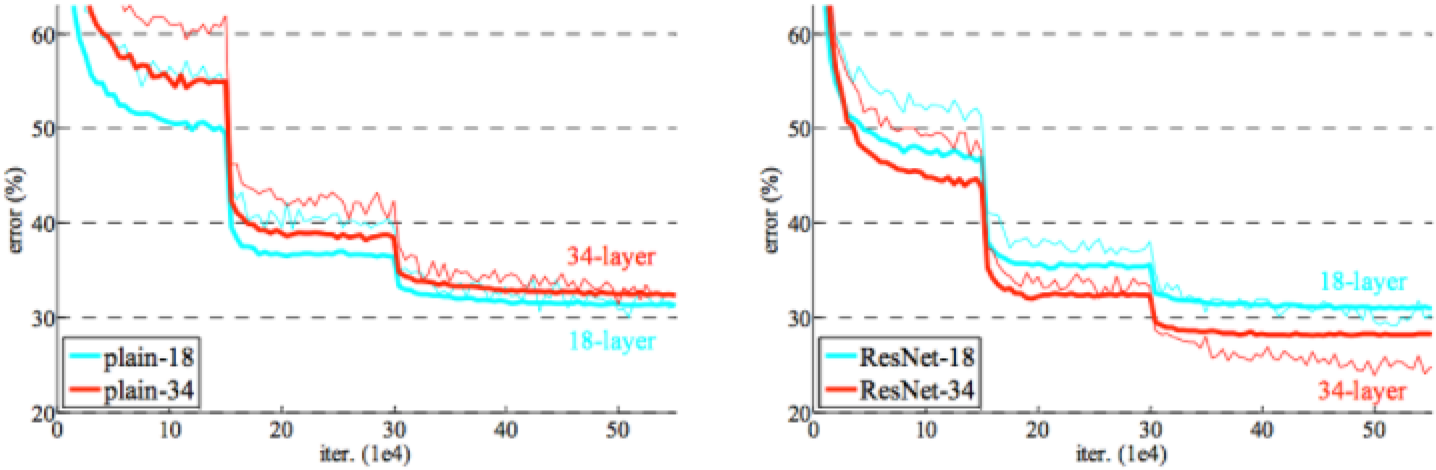
(a) original



```
shorcut = x

out = self.conv1(x)
out = self.bn1(out)
out = self.relu(out)

out = self.conv2(out)
out = self.bn2(out)

out += shortcut
out = self.relu(out)
```

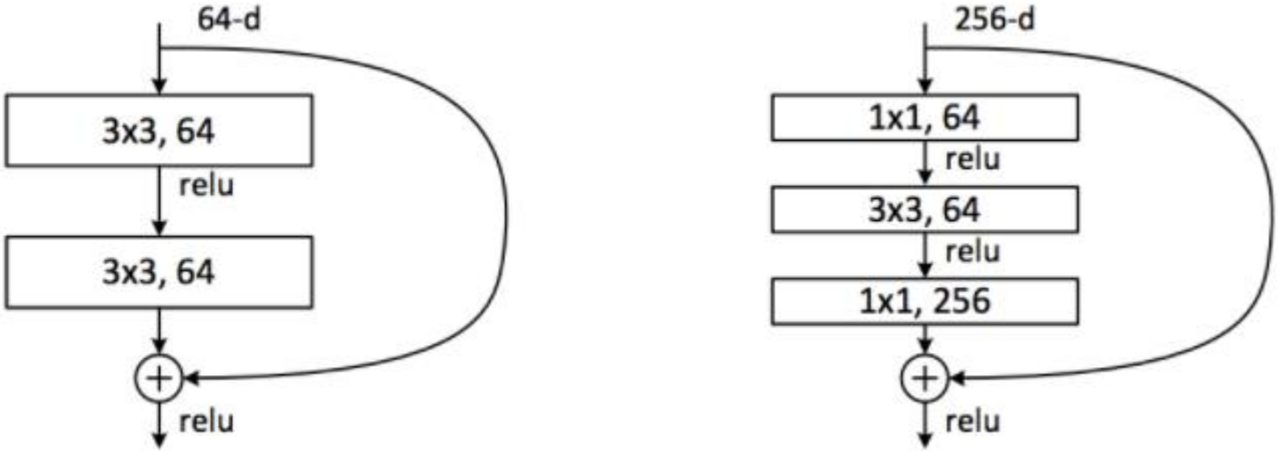
ResNet

- PlainNet vs. ResNet



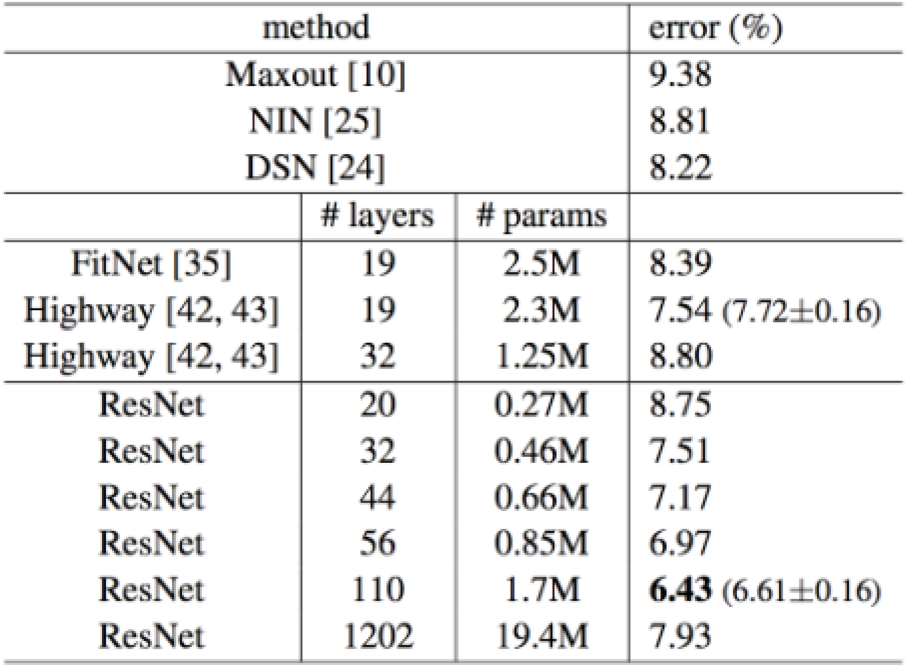
ResNet

- Bottle Neck



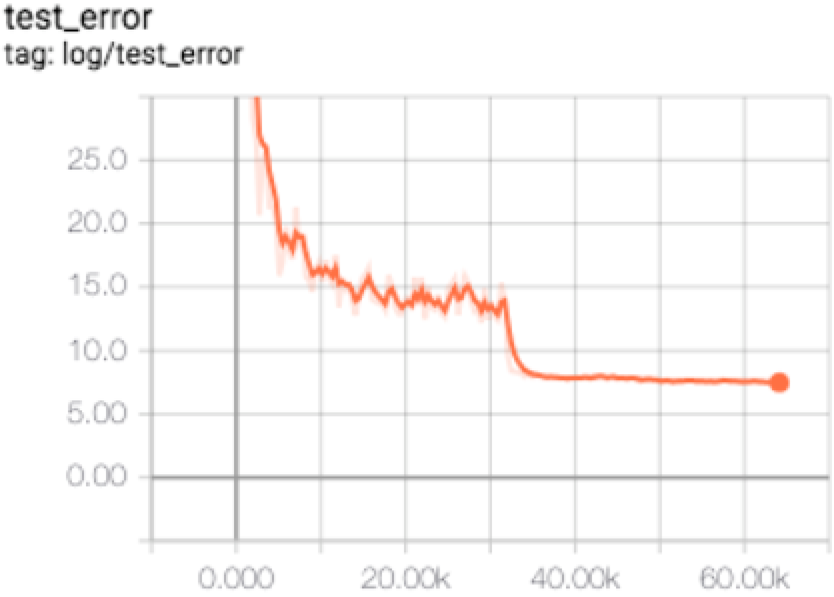
ResNet

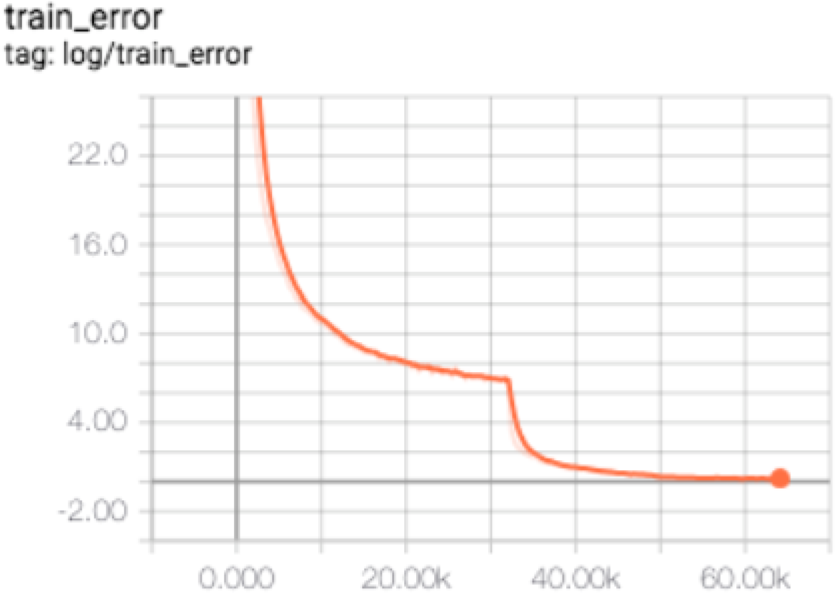
- Limitation



method			error (%)
Maxout [10]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [35]	19	2.5M	8.39
Highway [42, 43]	19	2.3M	7.54 (7.72±0.16)
Highway [42, 43]	32	1.25M	8.80
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	**6.43** (6.61±0.16)
ResNet	1202	19.4M	7.93

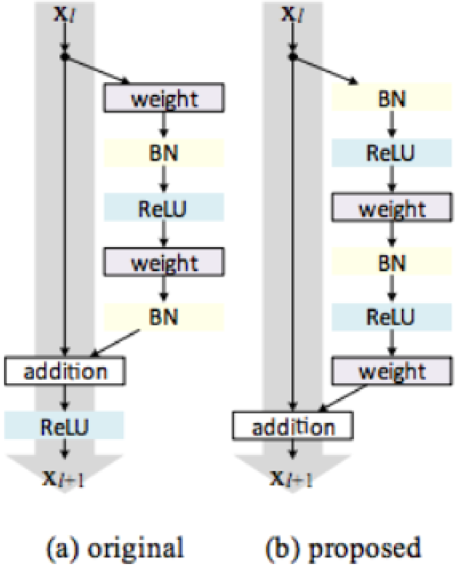
ResNet

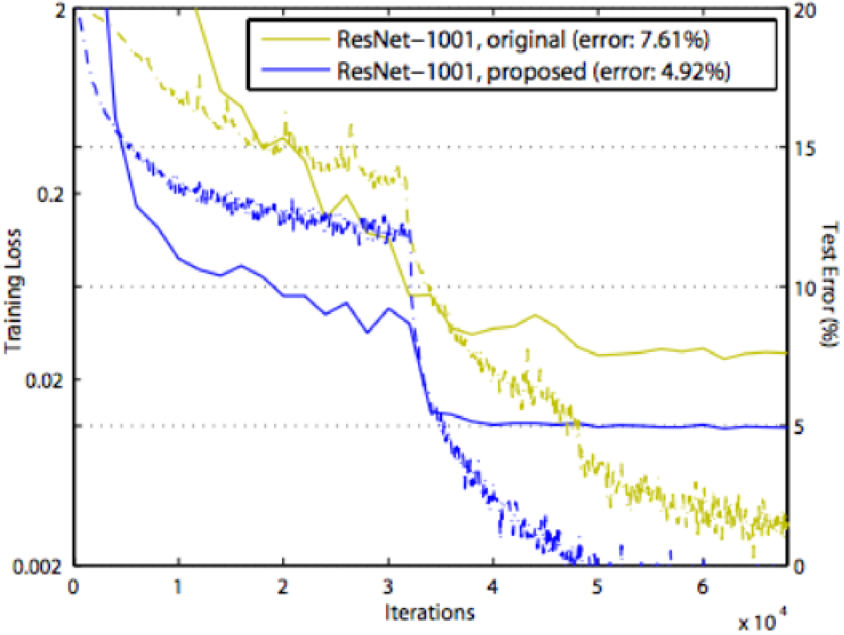

test_error
tag: log/test_error


train_error
tag: log/train_error

From 100 to 1000 Layers

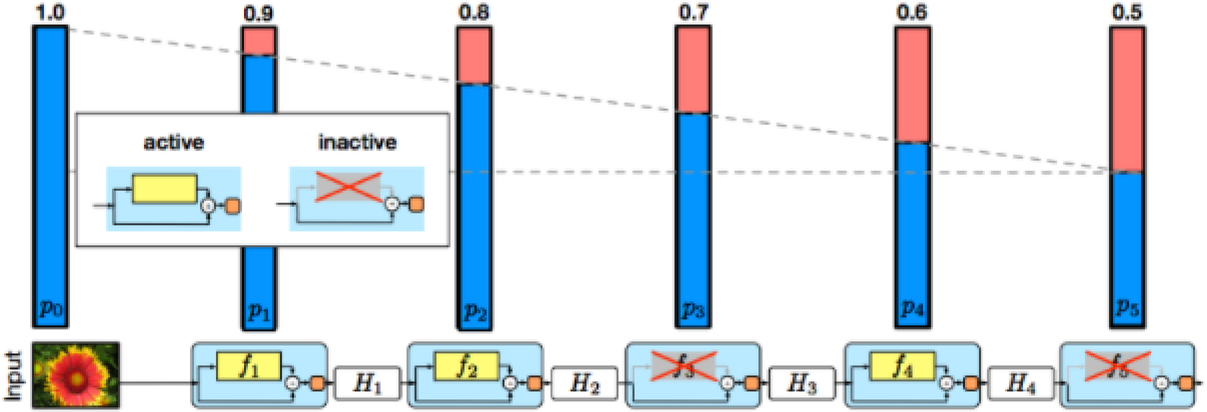
- pre-activation





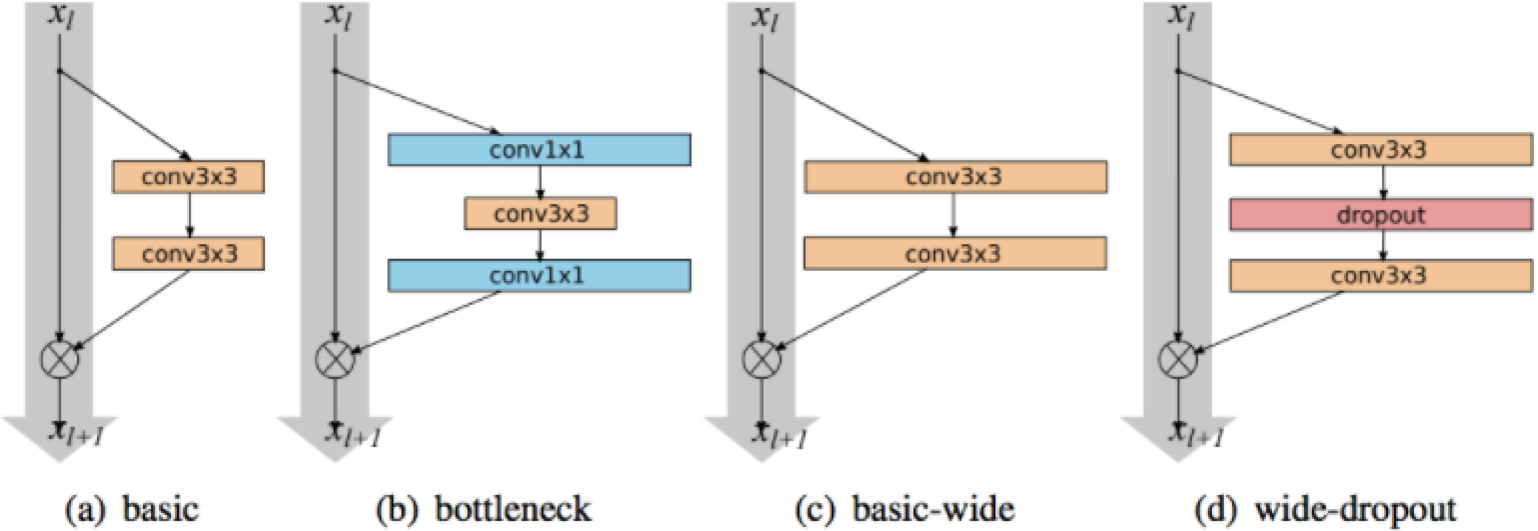
WideResNet

- ResDrop



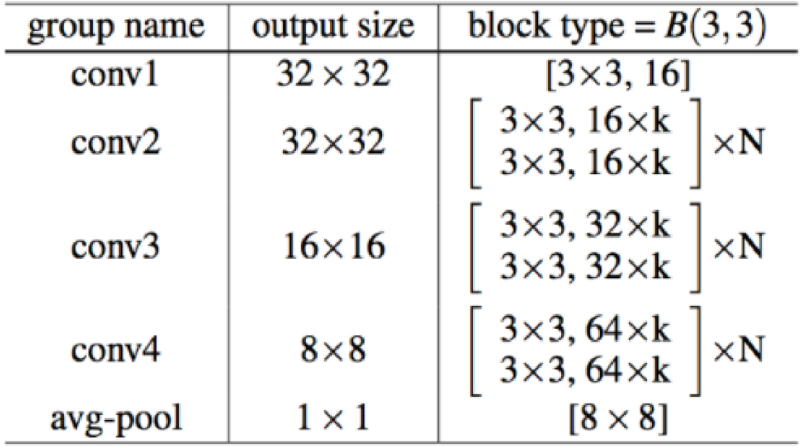
WideResNet

- Wide

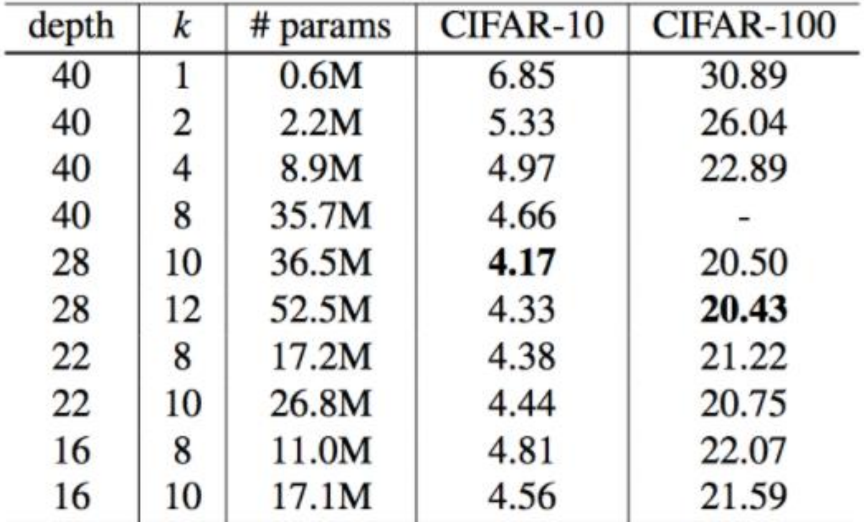


WideResNet

- Wide



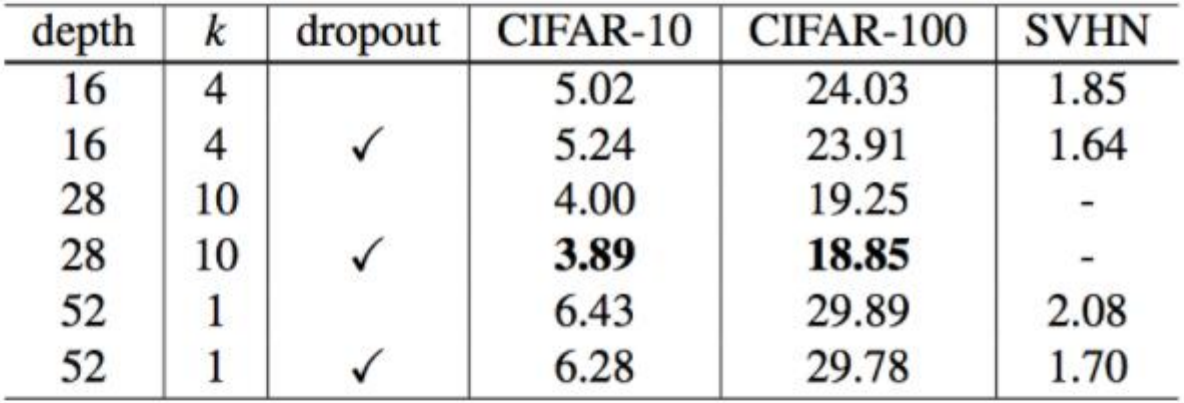
group name	output size	block type = $B(3,3)$
conv1	32×32	[3×3, 16]
conv2	32×32	$\begin{bmatrix} 3\times3, 16\times k \\ 3\times3, 16\times k \end{bmatrix} \times N$
conv3	16×16	$\begin{bmatrix} 3\times3, 32\times k \\ 3\times3, 32\times k \end{bmatrix} \times N$
conv4	8×8	$\begin{bmatrix} 3\times3, 64\times k \\ 3\times3, 64\times k \end{bmatrix} \times N$
avg-pool	1×1	$[8 \times 8]$



depth	*k*	# params	CIFAR-10	CIFAR-100
40	1	0.6M	6.85	30.89
40	2	2.2M	5.33	26.04
40	4	8.9M	4.97	22.89
40	8	35.7M	4.66	-
28	10	36.5M	**4.17**	20.50
28	12	52.5M	4.33	**20.43**
22	8	17.2M	4.38	21.22
22	10	26.8M	4.44	20.75
16	8	11.0M	4.81	22.07
16	10	17.1M	4.56	21.59

WideResNet

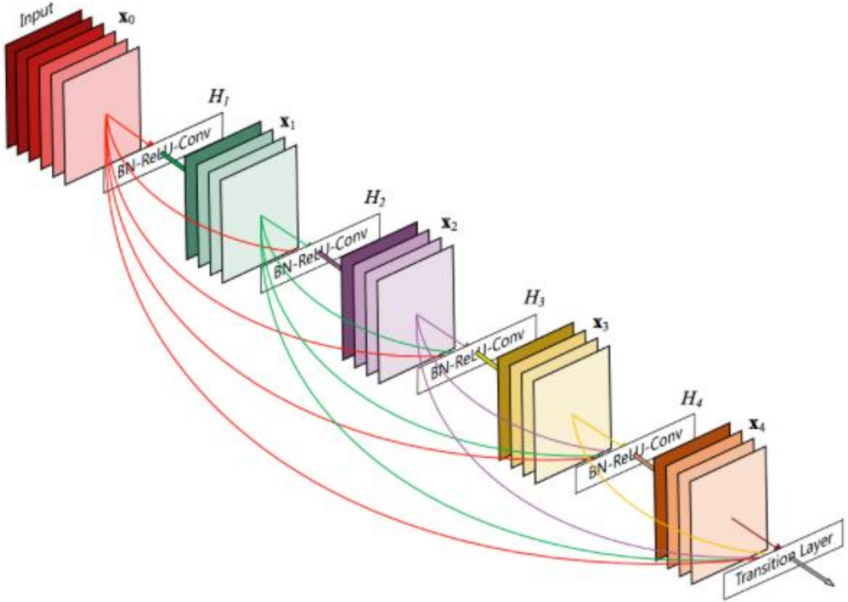
- Wide



depth	k	dropout	CIFAR-10	CIFAR-100	SVHN
16	4		5.02	24.03	1.85
16	4	✓	5.24	23.91	1.64
28	10		4.00	19.25	-
28	10	✓	**3.89**	**18.85**	-
52	1		6.43	29.89	2.08
52	1	✓	6.28	29.78	1.70

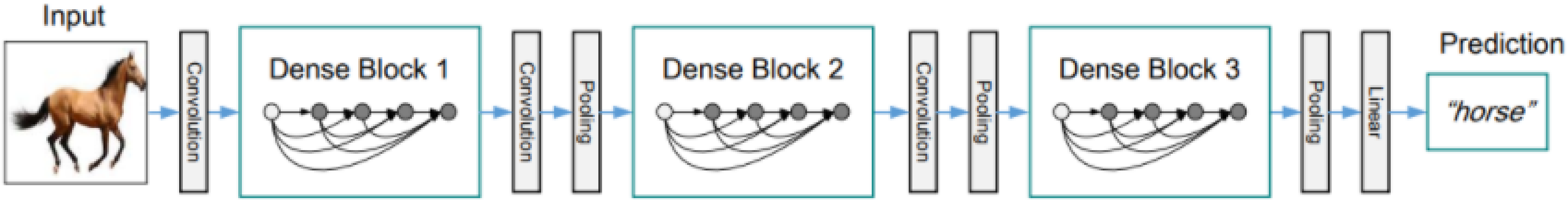
DenseNet

- DenseNet



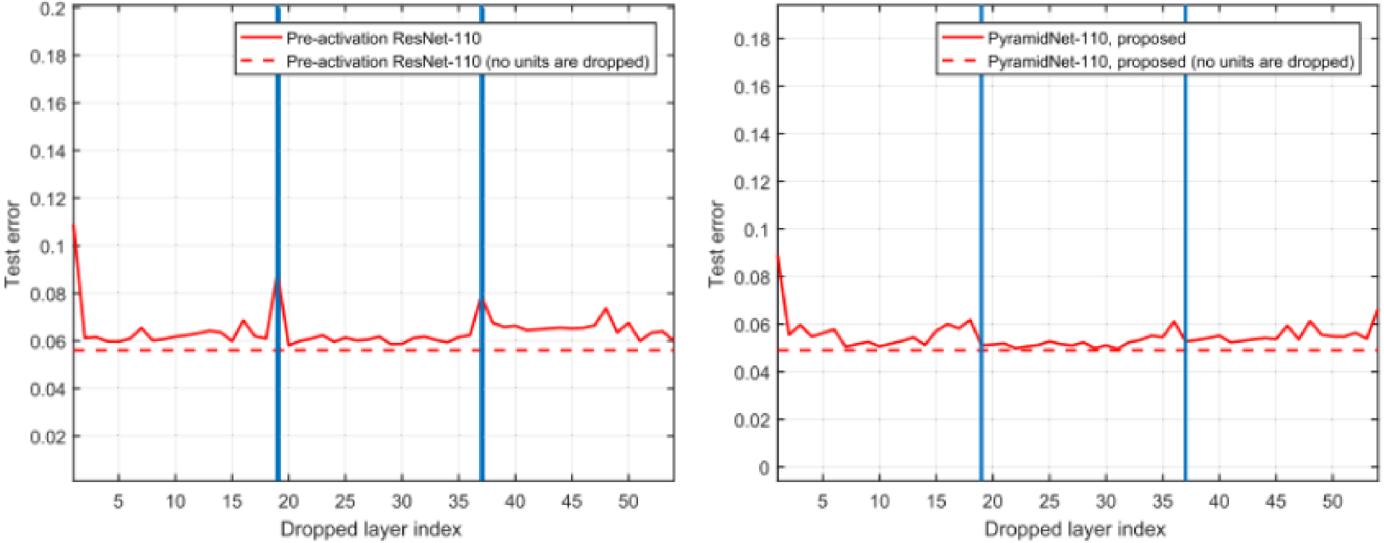
DenseNet

- DenseNet



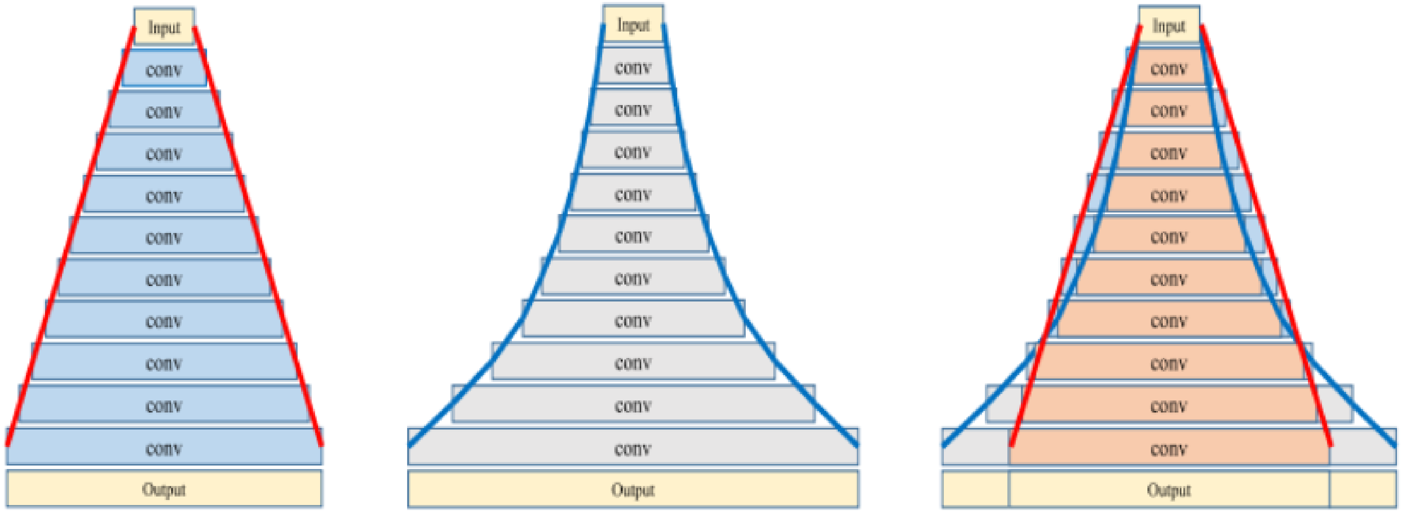
PyramidNet

- PyramidNet : ResNet에서의 down sampling 해결



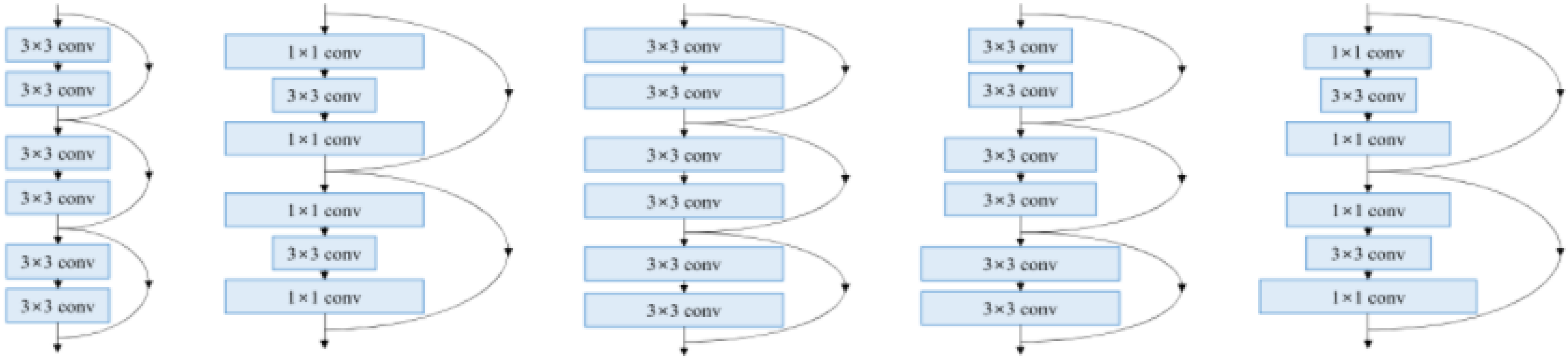
PyramidNet

- Why ‘Pyramid’?



PyramidNet

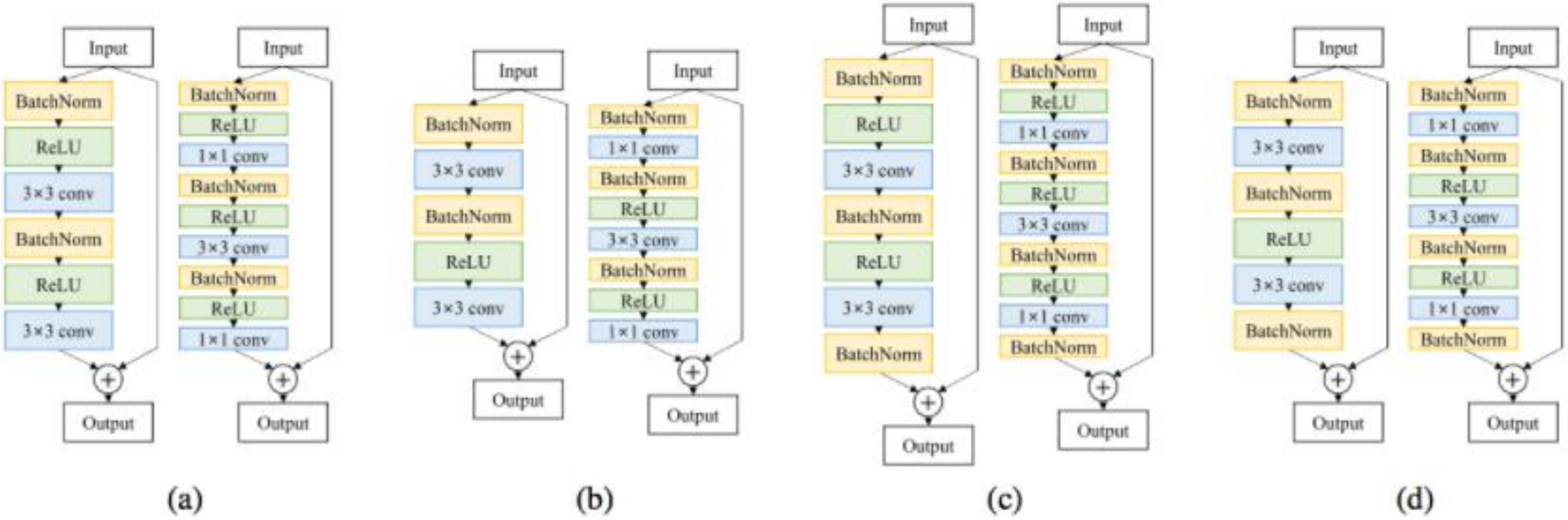
- PyramidNet's Residual Block



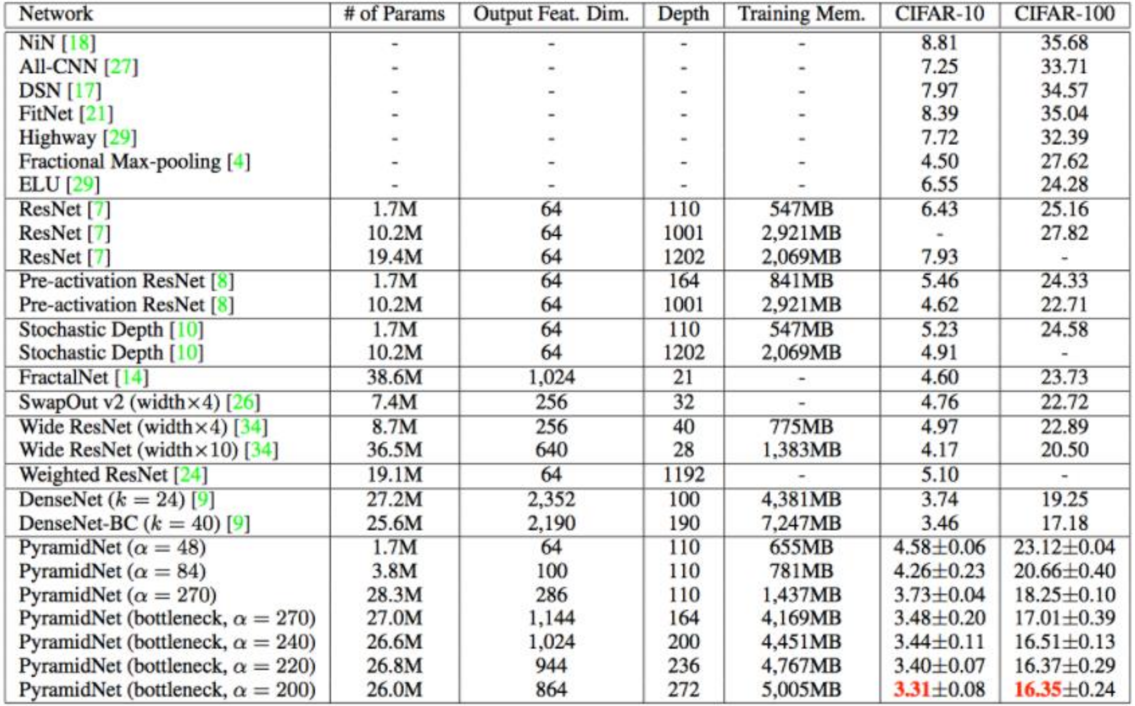
(a) basic (b) bottleneck (c) wide (d) pyramidal (e) pyramidal bottleneck

PyramidNet

- PyramidNet's Residual Block

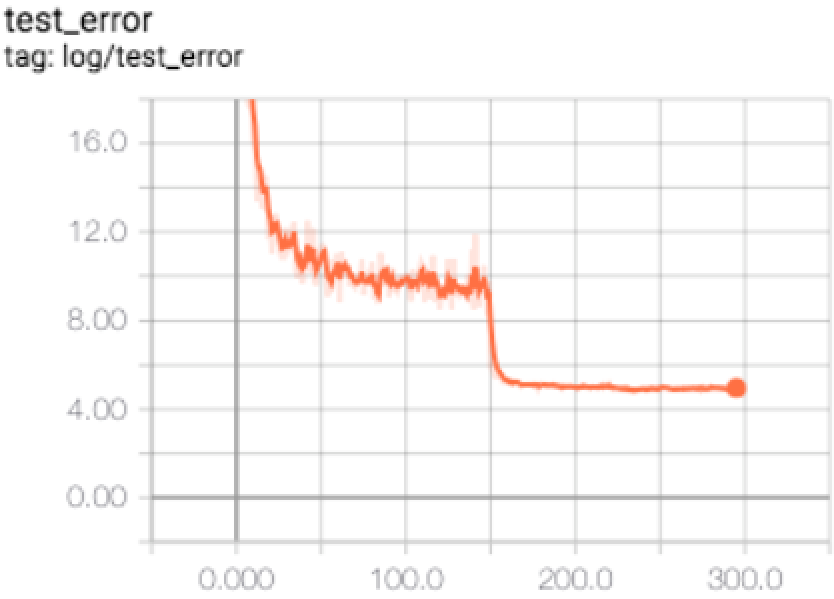


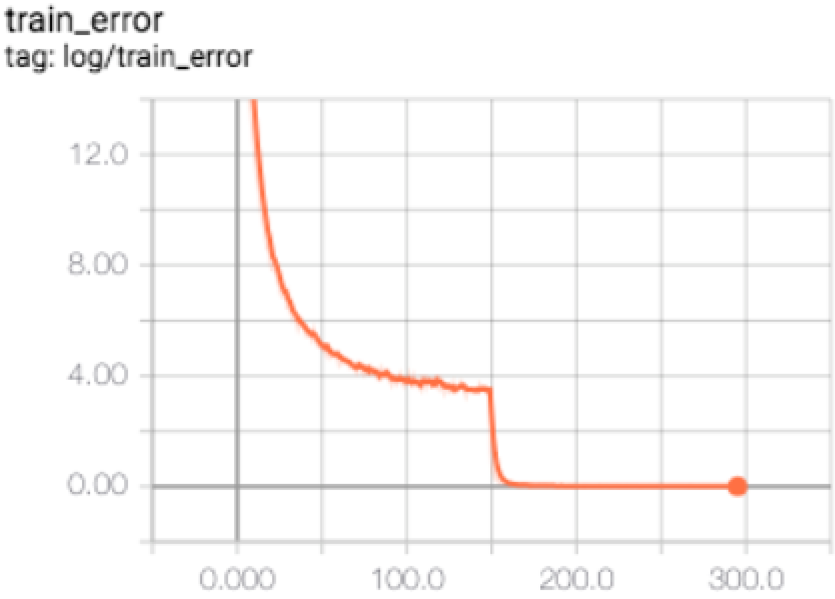
PyramidNet



Network	# of Params	Output Feat. Dim.	Depth	Training Mem.	CIFAR-10	CIFAR-100
NiN [18]	-	-	-	-	8.81	35.68
All-CNN [27]	-	-	-	-	7.25	33.71
DSN [17]	-	-	-	-	7.97	34.57
FitNet [21]	-	-	-	-	8.39	35.04
Highway [29]	-	-	-	-	7.72	32.39
Fractional Max-pooling [4]	-	-	-	-	4.50	27.62
ELU [29]	-	-	-	-	6.55	24.28
ResNet [7]	1.7M	64	110	547MB	6.43	25.16
ResNet [7]	10.2M	64	1001	2,921MB	-	27.82
ResNet [7]	19.4M	64	1202	2,069MB	7.93	-
Pre-activation ResNet [8]	1.7M	64	164	841MB	5.46	24.33
Pre-activation ResNet [8]	10.2M	64	1001	2,921MB	4.62	22.71
Stochastic Depth [10]	1.7M	64	110	547MB	5.23	24.58
Stochastic Depth [10]	10.2M	64	1202	2,069MB	4.91	-
FractalNet [14]	38.6M	1,024	21	-	4.60	23.73
SwapOut v2 (width×4) [26]	7.4M	256	32	-	4.76	22.72
Wide ResNet (width×4) [34]	8.7M	256	40	775MB	4.97	22.89
Wide ResNet (width×10) [34]	36.5M	640	28	1,383MB	4.17	20.50
Weighted ResNet [24]	19.1M	64	1192	-	5.10	-
DenseNet ($k = 24$) [9]	27.2M	2,352	100	4,381MB	3.74	19.25
DenseNet-BC ($k = 40$) [9]	25.6M	2,190	190	7,247MB	3.46	17.18
PyramidNet ($\alpha = 48$)	1.7M	64	110	655MB	4.58±0.06	23.12±0.04
PyramidNet ($\alpha = 84$)	3.8M	100	110	781MB	4.26±0.23	20.66±0.40
PyramidNet ($\alpha = 270$)	28.3M	286	110	1,437MB	3.73±0.04	18.25±0.10
PyramidNet (bottleneck, $\alpha = 270$)	27.0M	1,144	164	4,169MB	3.48±0.20	17.01±0.39
PyramidNet (bottleneck, $\alpha = 240$)	26.6M	1,024	200	4,451MB	3.44±0.11	16.51±0.13
PyramidNet (bottleneck, $\alpha = 220$)	26.8M	944	236	4,767MB	3.40±0.07	16.37±0.29
PyramidNet (bottleneck, $\alpha = 200$)	26.0M	864	272	5,005MB	**3.31**±0.08	**16.35**±0.24

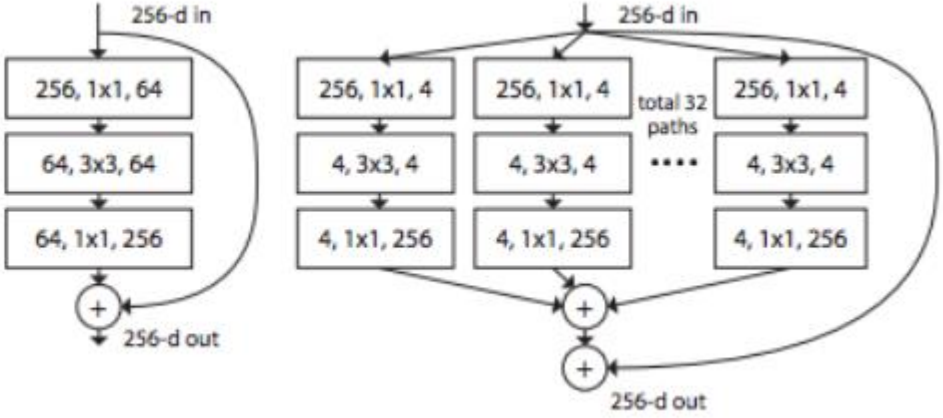
PyramidNet

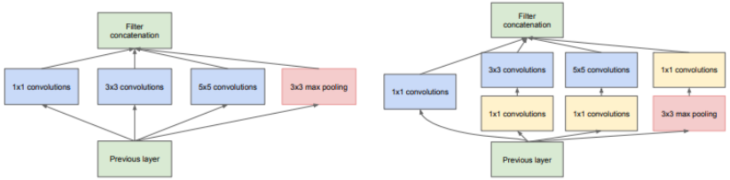

test_error
tag: log/test_error
16.0
12.0
8.00
4.00
0.00
0.000
100.0
200.0
300.0


train_error
tag: log/train_error
12.0
8.00
4.00
0.00
0.000
100.0
200.0
300.0

Shake-Shake

- ResNeXt



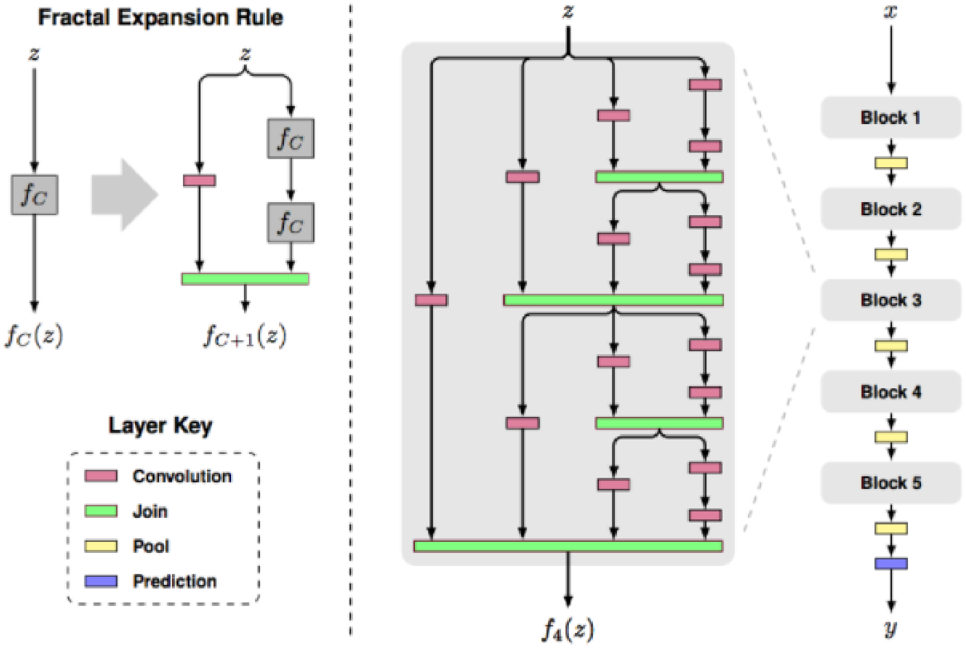


(a) Inception module, naïve version

(b) Inception module with dimension reductions

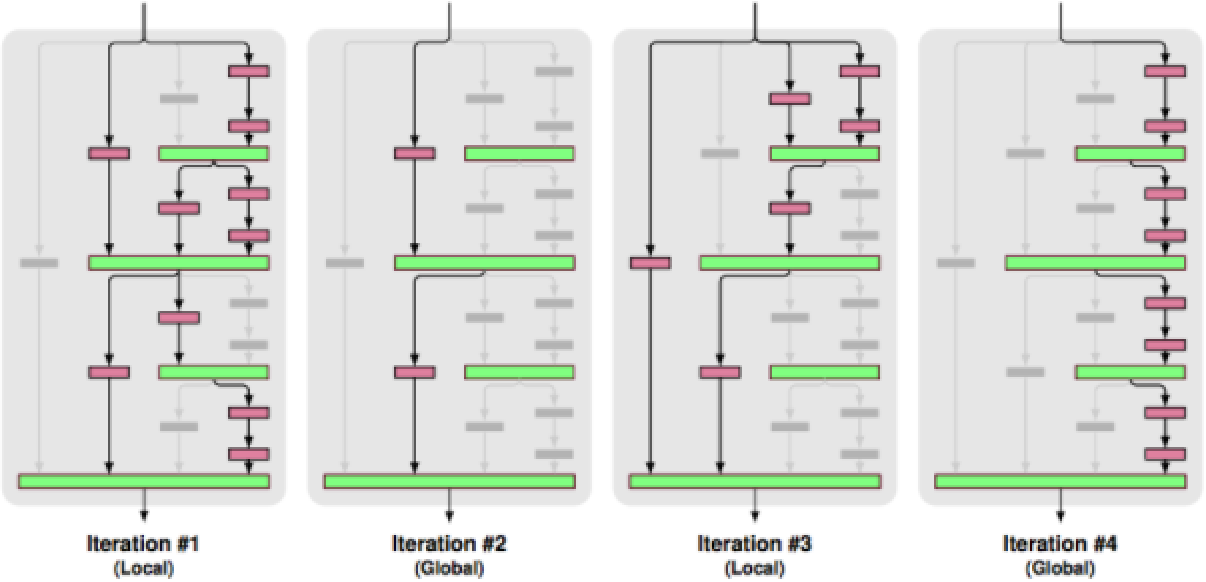
FractalNet

- FractalNet



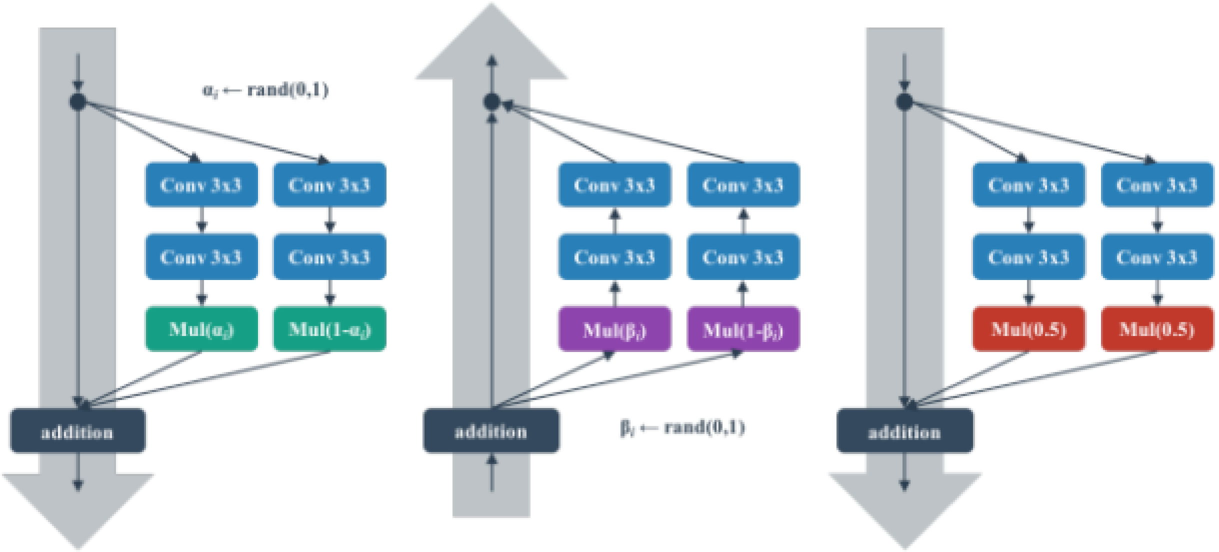
FractalNet

- FractalNet



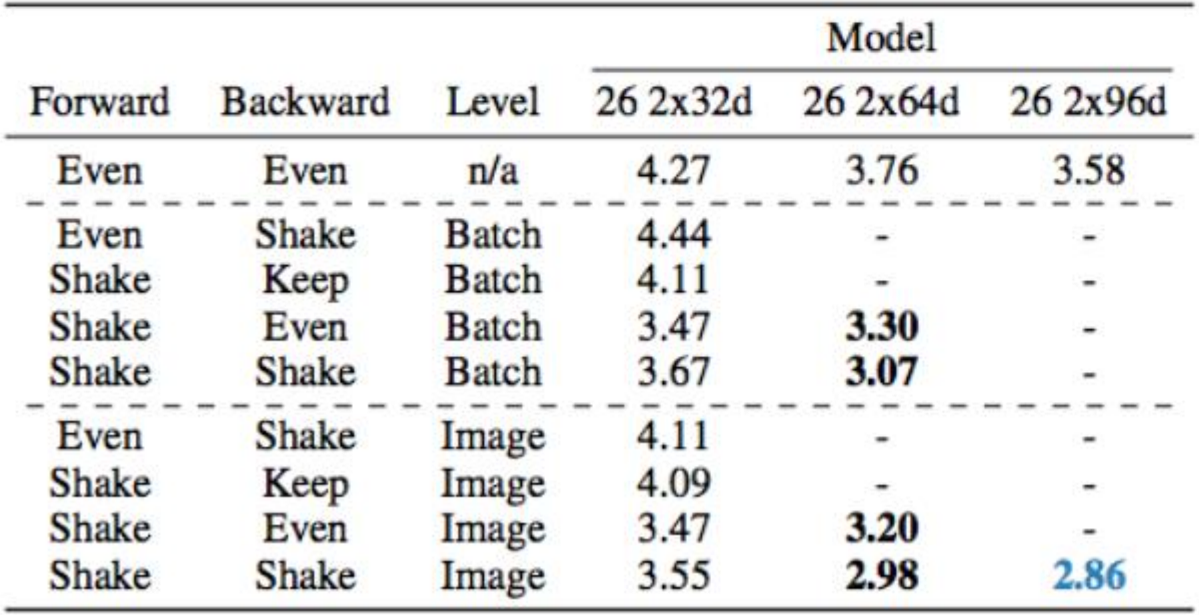
Shake-Shake

- Shake-Shake



Shake-Shake

- Shake-Shake



			Model		
Forward	Backward	Level	26 2x32d	26 2x64d	26 2x96d
Even	Even	n/a	4.27	3.76	3.58
Even	Shake	Batch	4.44	-	-
Shake	Keep	Batch	4.11	-	-
Shake	Even	Batch	3.47	**3.30**	-
Shake	Shake	Batch	3.67	**3.07**	-
Even	Shake	Image	4.11	-	-
Shake	Keep	Image	4.09	-	-
Shake	Even	Image	3.47	**3.20**	-
Shake	Shake	Image	3.55	**2.98**	**2.86**

Shake-Shake

- Cosine annealing

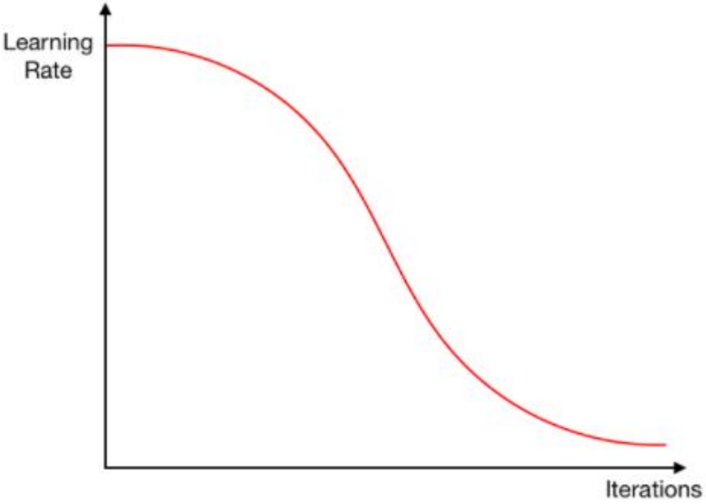
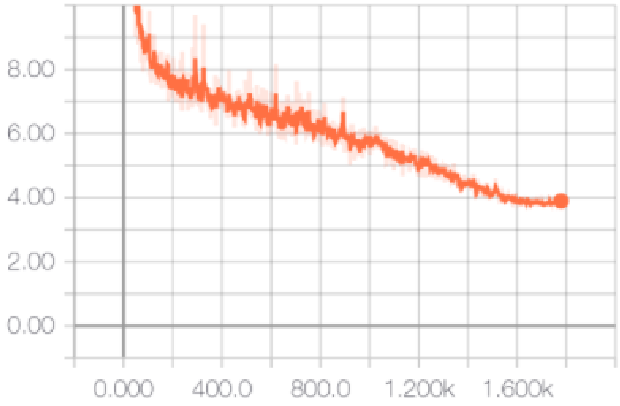


Image 4: Cosine Annealing

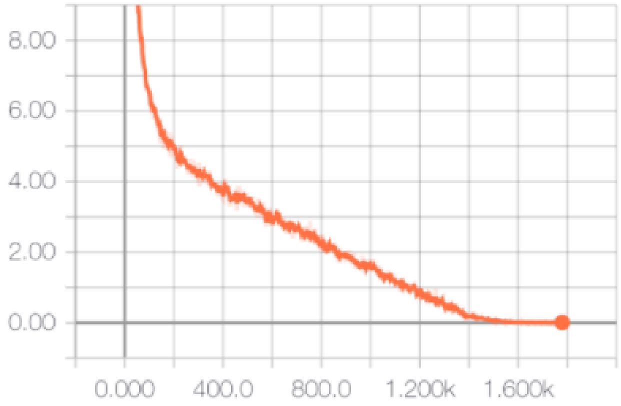
Shake-Shake

- Result

test_error
tag: log/test_error



train_error
tag: log/train_error


8.00
6.00
4.00
2.00
0.00
0.000
400.0
800.0
1.200k
1.600k