Identity Mappings in Deep Residual Networks

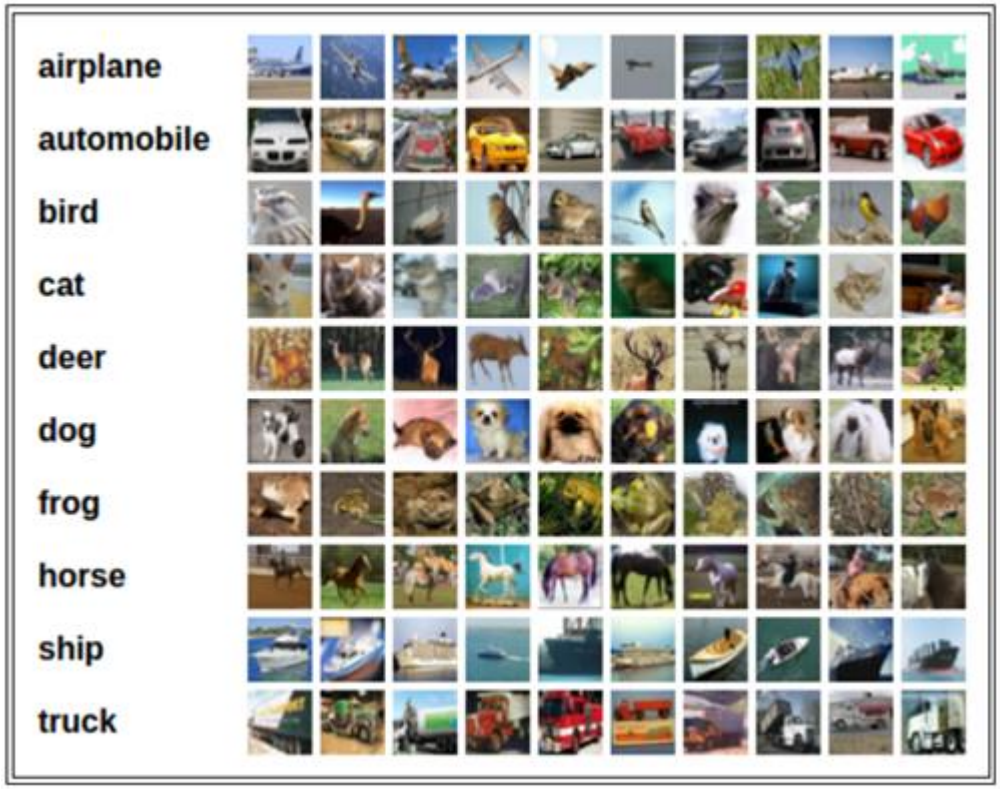
Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun
Microsoft Research

Paper review

KAERI-UST
Jungmin Kim

CIFAR-10

- Image classification dataset 중에서 CIFAR-10 데이터
- 32 x 32 픽셀 크기의 60000 개의 작은 컬러 이미지(3X32X32)
- 학습데이터 50000개 테스트 데이터 10000개로 구성



Deep Residual Learning for Image Recognition

- CNN에서 네트워크의 층(layer)을 더 쌓으며 아주 깊은 네트워크를 쌓으면서 좋은 성능 보임
- 어느정도 이상 깊어진 네트워크는 오히려 성능이 떨어지는 Degradation problem (vanishing/exploding gradient)
- 깊은 네트워크에서 Degradation을 해결할 방법으로 residual learning을 적용한 **ResNet** 제안

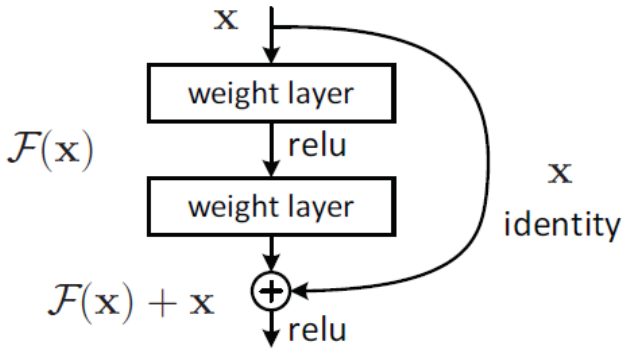
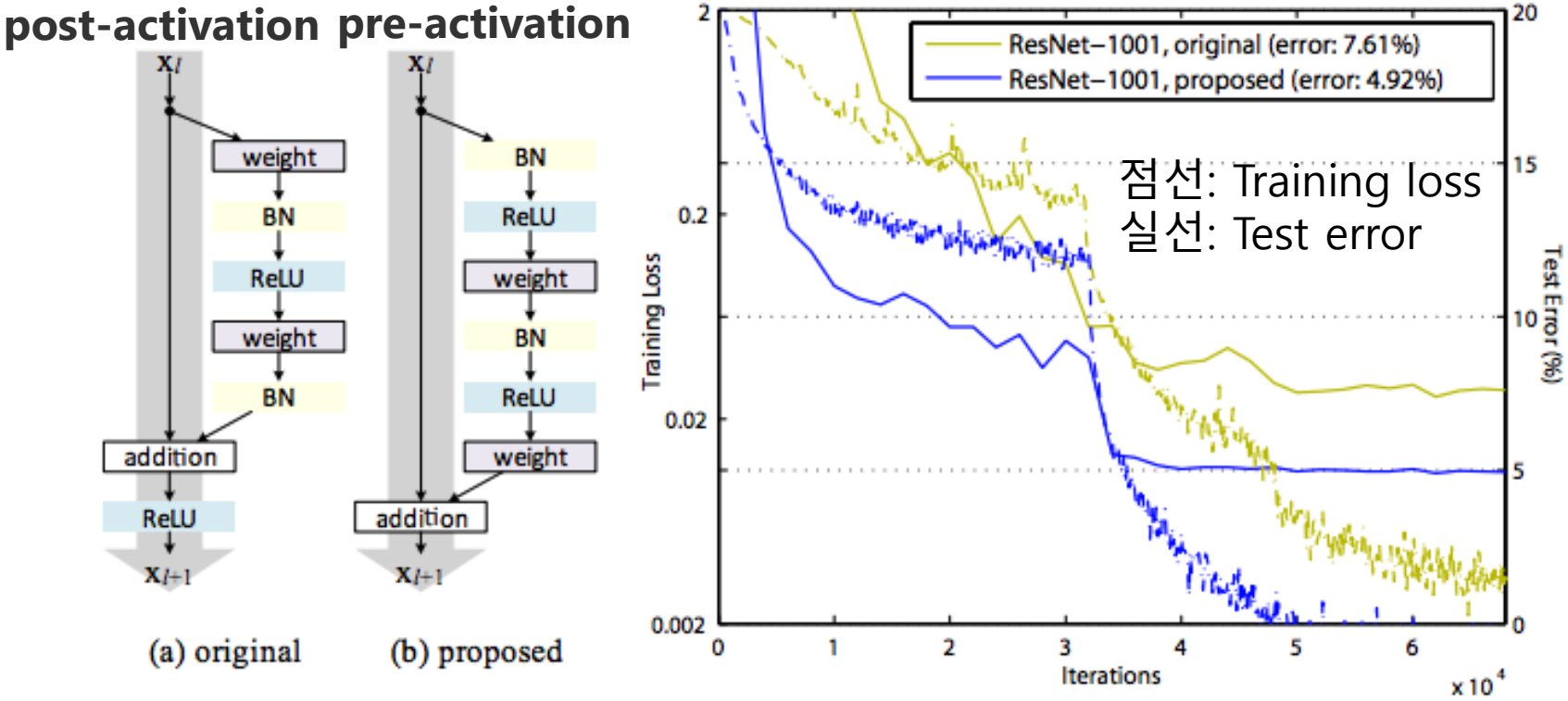


Figure 2. Residual learning: a building block.

- ResNet은 진정한 Deep Learning의 시대를 연 영향력이 큰 네트워크

Identity Mappings in Deep Residual Networks

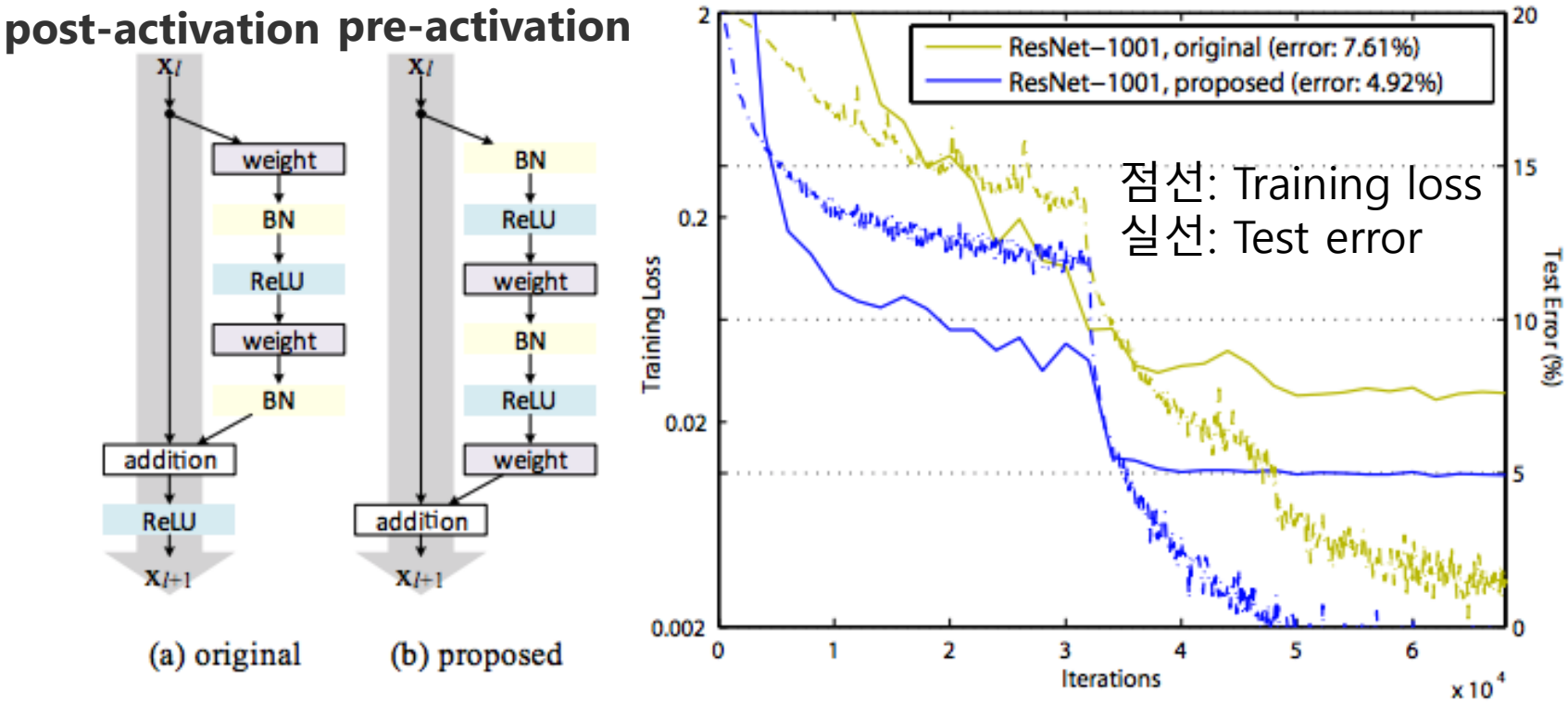
ResNet의 저자인 Kaiming He가 새로운 Residual block의 구조를 제안



Post-activation(original): conv3x3 + BN + ReLU + conv3x3 + BN→shortcut connection→ ReLU

Pre-activation(proposed): BN + ReLU + conv3x3 + BN + ReLU + conv3x3 →shortcut connection

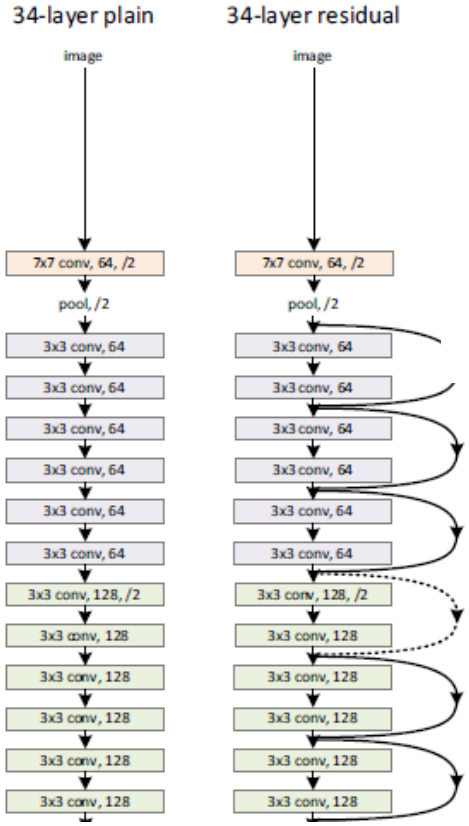
Identity Mappings in Deep Residual Networks

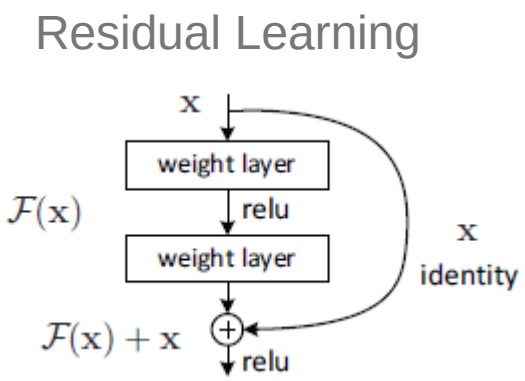


ResNet의 저자인 Kaiming He가 새로운 Residual block의 구조를 제안

Proposed ResNet much easier to train and generalizes better than the original ResNet.

Residual Learning의 효과





기존 **CNN**: H(x)의 최소화

ResNet: F(x) (= H(x) - x) 최소화→ H(x) = x

$$y_l = h(x_l) + \mathcal{F}(x_l, \mathcal{W}_l), \quad (1)$$

$$x_{l+1} = f(y_l). \quad (2)$$

$x_{l+1} = y_l$

$$x_{l+1} = x_l + \mathcal{F}(x_l, \mathcal{W}_l). \quad (3)$$

$$x_L = x_l + \sum_{i=l}^{L-1} \mathcal{F}(x_i, \mathcal{W}_i), \quad (4)$$

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1}\mathcal{F}(x_i, \mathcal{W}_i)\right). \quad (5)$$

Residual Unit을 사용한다면 **네트워크**를 Residual Function인 F의 합으로 표현

$$\prod_{i=0}^{L-1} W_i x_0$$

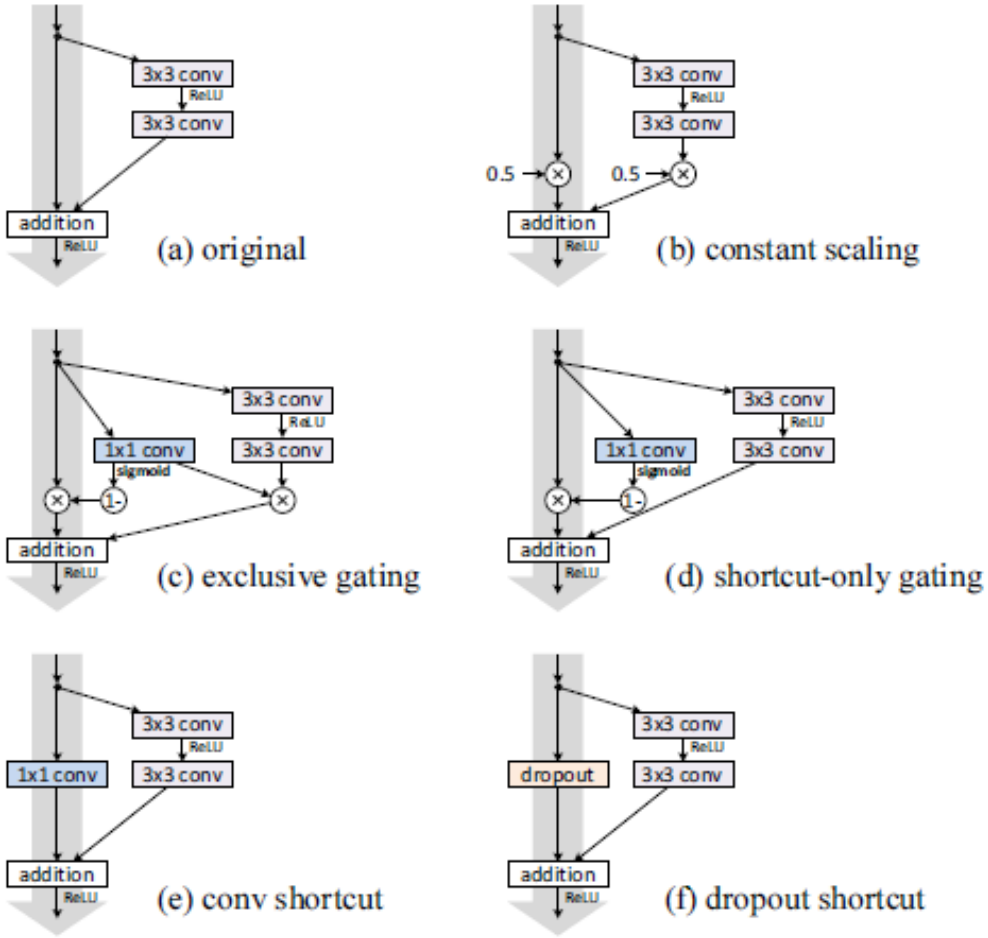
행렬들의 곱셈으로 표현되는 일반 네트워크에 비해 가중치 행렬들의 곱셈으로 정보가 전달되는 것이 아니기 때문에 Vanishing 문제가 발생하지 않는다.

Feed Forwarding: Residual Unit의 Shortcut을 이용해 곱셈이 아닌 덧셈으로 쉬운 정보이동이 가능

Backpropagation: 수많은 레이어의 가중치 행렬들의 곱해지지 않으므로 Vanishing 문제를 막을수 있음

→ Residual Unit의 Shortcut을 이용해 네트워크의 깊이의 한계를 극복

Residual Unit의 구조에 따른 차이



CIFAR-10의 "Deep Residual Learning for Image Recognition"에 제시된 110-layer ResNet으로 실험

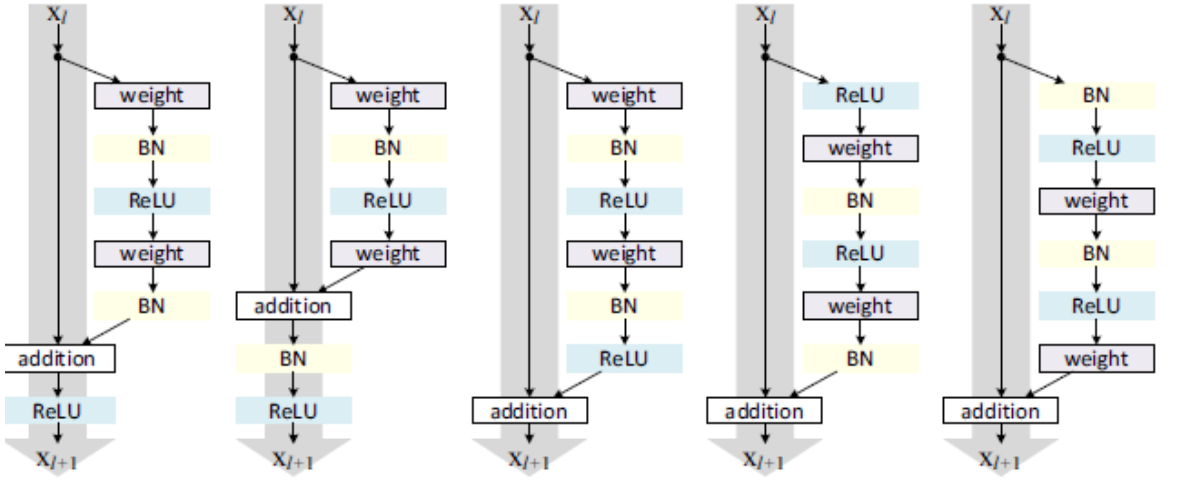
case	Fig.	on shortcut	on $\mathcal{F}$	error (%)	remark
original [1]	Fig. 2(a)	1	1	**6.61**	
constant scaling	Fig. 2(b)	0	1	fail	This is a plain net
		0.5	1	fail	
		0.5	0.5	12.35	frozen gating
exclusive gating	Fig. 2(c)	$1-g(\mathbf{x})$	$g(\mathbf{x})$	fail	init b_g=0 to −5
		$1-g(\mathbf{x})$	$g(\mathbf{x})$	8.70	init b_g=-6
		$1-g(\mathbf{x})$	$g(\mathbf{x})$	9.81	init b_g=-7
shortcut-only gating	Fig. 2(d)	$1-g(\mathbf{x})$	1	12.86	init b_g=0
		$1-g(\mathbf{x})$	1	6.91	init b_g=-6
1×1 conv shortcut	Fig. 2(e)	1×1 conv	1	12.22	
dropout shortcut	Fig. 2(f)	dropout 0.5	1	fail	

"fail": test error>20%

다양한 종류의 shortcut connection들은 결과가 수렴하지 못하거나, 연산 요소들(Multiplicative manipulations)에 의한 정보 전달 방해로 최적화 문제

→Original residual unit의 test error가 가장 낮았다.

Activation Functions에 따른 차이



case	Fig.	ResNet-110	ResNet-164
original Residual Unit [1]	Fig. 4(a)	6.61	5.93
BN after addition	Fig. 4(b)	8.17	6.50
ReLU before addition	Fig. 4(c)	7.84	6.14
ReLU-only pre-activation	Fig. 4(d)	6.71	5.91
full pre-activation	Fig. 4(e)	**6.37**	**5.46**

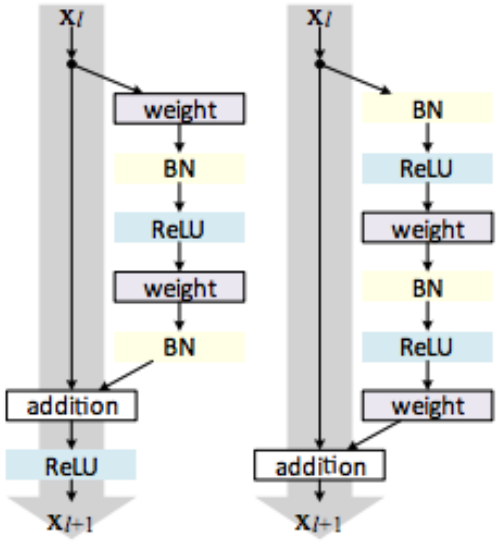
Classification error (%) on the CIFAR-10 test set using different activation functions.

CIFAR-10 데이터를 사용했으며 BN(Batch Normalization)과 ReLU를 다양한 시점에 적용

Addition 단계 전에 Activation을 적용하는 'Pre-Activation' (e)가 가장 좋은 결과

Activation Functions에 따른 차이

post-activation pre-activation



(a) original (b) proposed

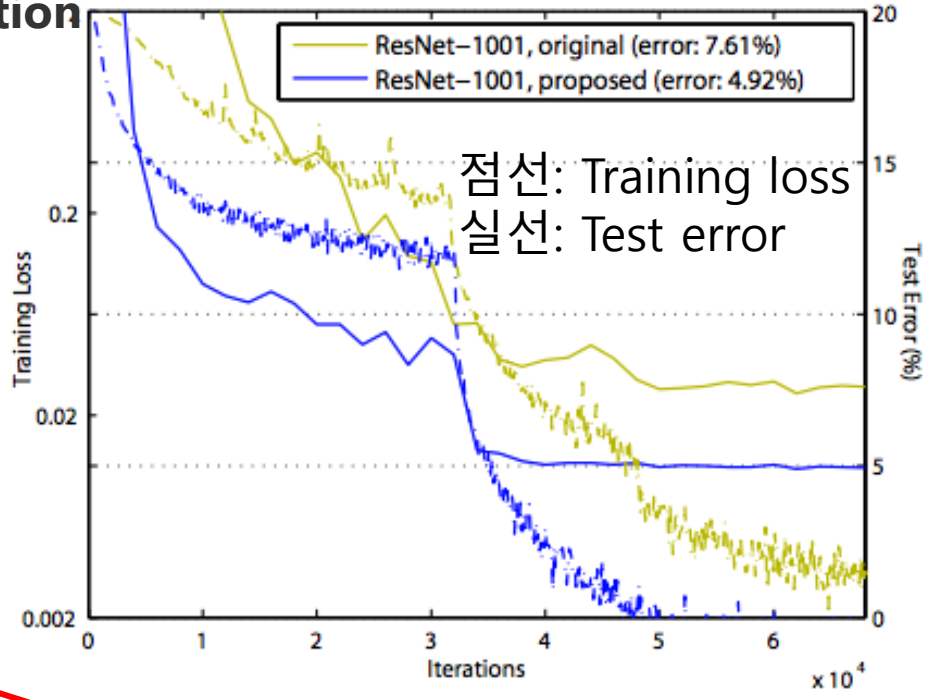
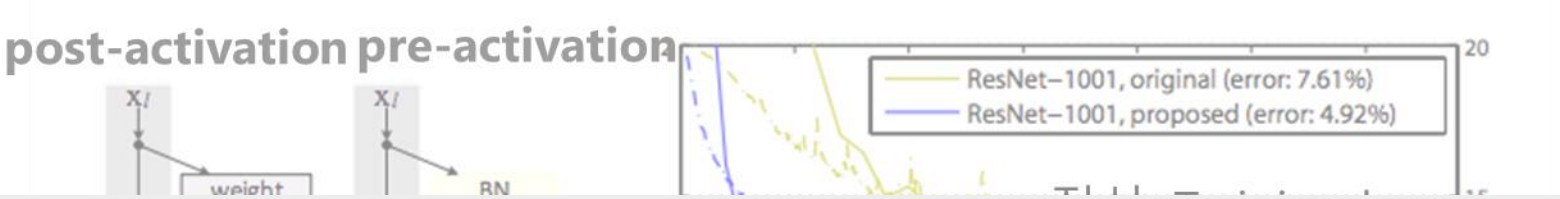


Table 3. Classification error (%) on the CIFAR-10/100 test set using the original Residual Units and our pre-activation Residual Units.

dataset	network	baseline unit	pre-activation unit
CIFAR-10	ResNet-110 (1layer skip)	9.90	8.91
	ResNet-110	6.61	6.37
	ResNet-164	5.93	5.46
	ResNet-1001	7.61	4.92
CIFAR-100	ResNet-164	25.16	24.33
	ResNet-1001	27.82	22.71

Activation Functions에 따른 차이



pre-activation을 사용한 ResNet이

기존의 Post-activation을 사용한 ResNet보다 우수함

<Analysis>

Ease of optimization

Reducing overfitting

CIFAR-10	ResNet-110	6.61	6.37
	ResNet-164	5.93	5.46
	ResNet-1001	7.61	4.92
CIFAR-100	ResNet-164	25.16	24.33
	ResNet-1001	27.82	22.71

Conclusion

기존 ResNet 의미를 분석하고 더 개선된 모델을 제시

Shortcut 디자인과 Activation Function배열에 따른 ResNet을 비교하여

Pre-activation을 사용한 ResNet이 최적화와 overfitting 감소를 통해 깊은 네트워크에서 효과적인 ResNet을 제안

Reference

https://steemit.com/kr/@codingart/7-17-colabo-pytorch-cifar-10
https://arxiv.org/pdf/1603.05027.pdf
https://m.blog.naver.com/PostView.nhn?blogId=siniphia&logNo=221387516366&proxyReferer=https:%2F%2Fwww.google.com%2F
https://dnddnjs.github.io/cifar10/2018/10/09/resnet/
https://seing.tistory.com/46

Thank You