# Identity Mappings in Deep Residual Networks


## Abstract

본 논문에서는 Residual Block의 연산과정에 대해 분석하고, 이를 통해 새로운 Residual Block을 고안한다.


## Introduction

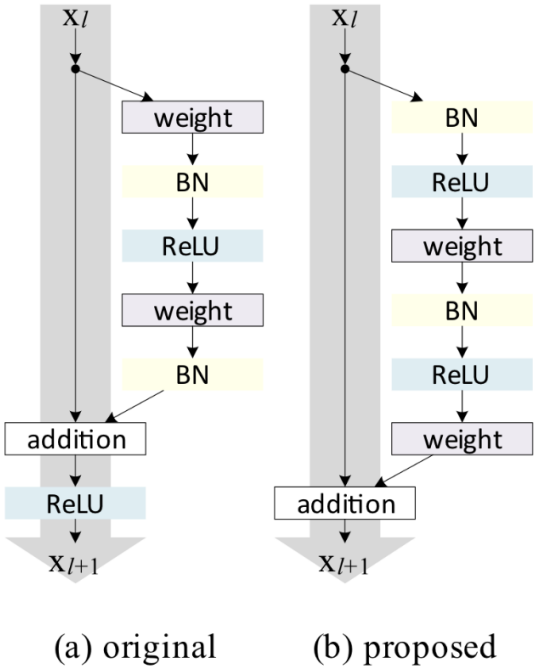



Fig 1. Residual Block

$$y_l = h(x_l) + F(x_l, W_l),$$

$$x_l + 1 = f(y_l)$$

여기서 $h(x_l)$는 identity mapping을 의미하며, $F(x_l, W_l)$은 residual mapping을 의미한다. 또한, $f(y_l)$은 ReLU activation을 적용시킨다는 의미다.

저자의 연구에 따르면, $h(x_l)$과 $f(y_l)$이 identity mapping일 때 forward 방향과 backward 방향 모두에서 데이터가 직접적으로 전달된다고 한다. 이 조건을 만족하는 구조가 Fig. 1(b) 이다.

또한, 기존 상식으로 여겨지던 “activation function은 Weight 연산 이후에 적용되어야 한다” 의 규칙을 Fig 1.(b) 처럼 깨버림으로써 학습이 더 잘 되고, 일반화도 더 잘되는 결과를 얻었다고 한다.

Activation function이 weight 연산 이전에 먼저 적용이 되어 모델의 이름을 “Pre-activation ResNet”으로 명명하였다.

## Anaylsis of Deep Residual Networks

위의 식에서 $h(x_l)$이 identity mapping 이라면 $h(x_l) = x_l$이고, $f(y_l)$이 identity mapping 이라면 $y_l = x_{l+1}$이 된다.

따라서, 식을 정리하면 다음과 같다.

$x_{l+1} = x_l + F(x_l, W_l)$

$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i)$

이 재귀식에서 중요한 성질들을 관찰할 수 있다.

1) 아무리 L이 깊더라도, 모델은 $x_l$과 $\sum_{i=l}^{L-1} F(x_i, W_i)$로 구성된다. 즉 L과 I 유닛들로 이루어진다.

2) $x_L = x_0 + \sum_{i=0}^{L-1} F(x_i, W_i)$이므로, $x_0$와 이전 모든 Residual function들의 합으로 볼 수 있다.

그렇다면, BackPropagation은 다음과 같이 표현된다.

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(1 + \frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)\right)$$

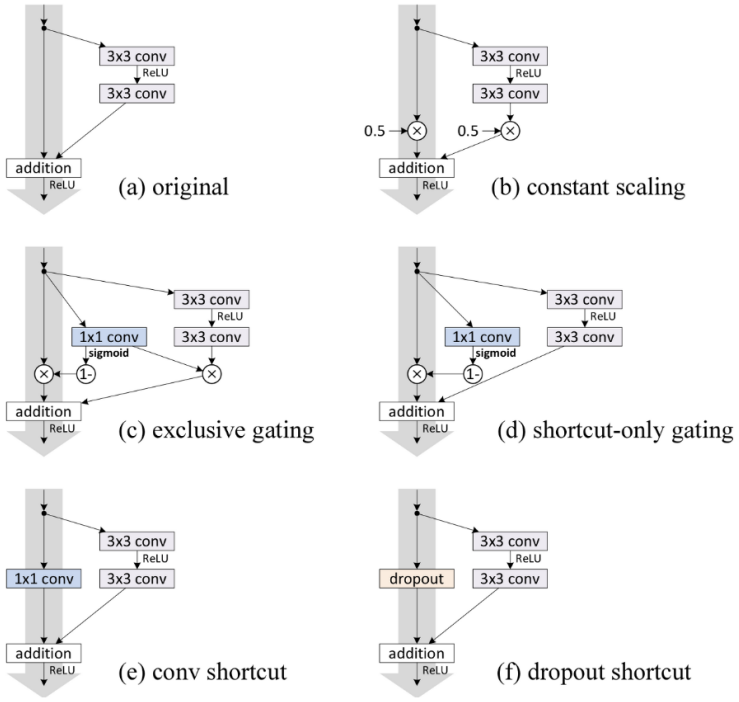
위 식을 보면, $\epsilon$에 대한 $x_l$의 편미분은 2개의 식의 합으로 나눠진다.

1) 전자식 : weight layer와는 관계없이 정보를 전달한다.

2) 후자식: weight layer를 통해서 정보가 전달된다.

## On the Importance of Identity Skip connections

이 부분에서는 $h(x_l)$이 identity mapping이 아니면 어떤 결과가 나올지에 대한 실험을 진행하였다.




1. Scalar : $h(x_l) = \lambda_l x_l$

- 스칼라 값을 하면, backpropagation시 스칼라 값이 중첩이 되어 곱해진다. 그 결과 $\lambda_l$이 1보다 작으면 Vanishing gradient problem이, 1보다 크면 Exploding gradient problem이 발생하게된다. 이는 최적화를 어렵게 만든다.

2. Exclusive gating

- 이는 간단히, 1x1 conv 연산 결과에 softmax를 적용시키고, 그 값을 이용하여 적절한 비율만큼 residual mapping과 identity mapping에 곱해주는 방식이다.
- shortcut에는 1-$g(x)$를 곱함
- 학습시 $g(x)$의 바이어스 $b_g$의 초기값에 영향을 크게 받음
- 0~-10 값 중 최적값을 찾아 활용
- $g(x)$가 0에 가까워지면 shortcut만 남아 identity-mapping에 가까워져 성능이 좋음

3. Shortcut only gating

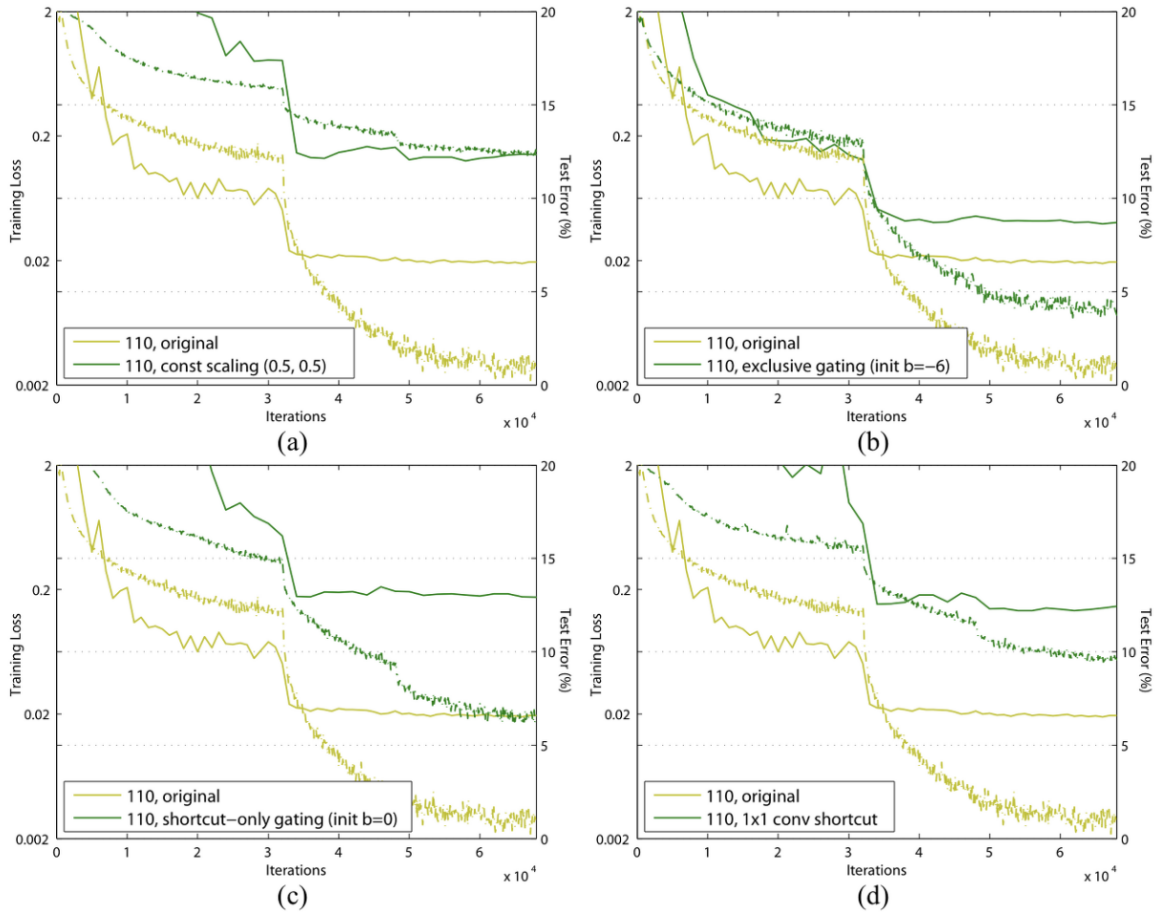
- 이는 Exclusive gating 과 유사하지만, residual mapping 부분은 softmax 출력값과 관계 없이 그대로 더해지는 방식이다.
- 2와 마찬가지로 g(x)가 0에 가까워지면 성능이 좋아짐

4. 1 x 1 Convolution

- 이는 identity mapping 을 1 x 1 shorcut connection으로 교체한 방식이다.
- ResNet 논문에서 projection shortcut과 동일
- ResNet 논문과는 다르게 항상 성능이 좋아지지는 않음
- 깊이에 따라 성능이 좋아지기도 나빠지기도 함

5. Dropout shortcut

- identity mapping에 dropout을 0.5만큼 적용시키는 방식이다. 이는 Scalar를 0.5로 둔 것과 유사하게 볼 수 있다.

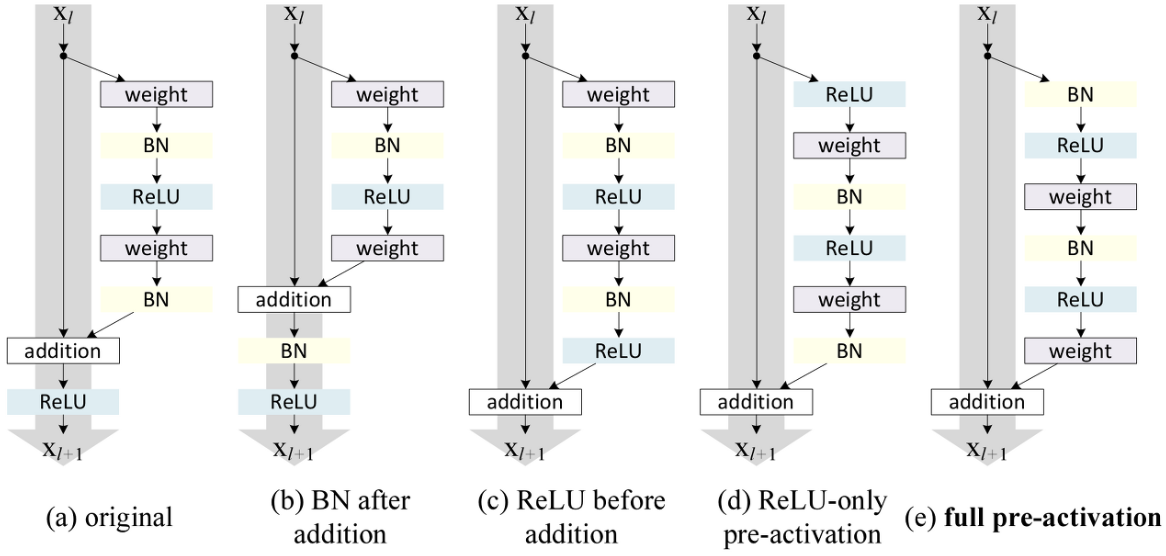


결과를 보면, 모든 경우 기존의 ResNet보다 성능이 좋지 않음을 알 수 있다.

## On the Usage of Activation Functions

이 부분에서는 $f(y_l)$이 identity mapping이 아니면 어떤 결과가 나오는지에 대한 실험을 진행

ResNet에서 사용하는 residual block에서 활성함수 f는 ReLU 함수이며, shortcut된 신호와 합쳐지는 곳 뒤에 위치한다. 논문에서는 f를 identity mapping으로 만들기 위해 새로운 residual block을 제안한다.




1. BN after addition // f : BN + ReLU

- 이 방식은 element-wise addition 이후에 BN을 넣고, ReLU를 적용시킨다.
  이 방식은 기존의 방식보다 성능이 안좋았다고 한다.

2. ReLU before addtion // f: identity mapping

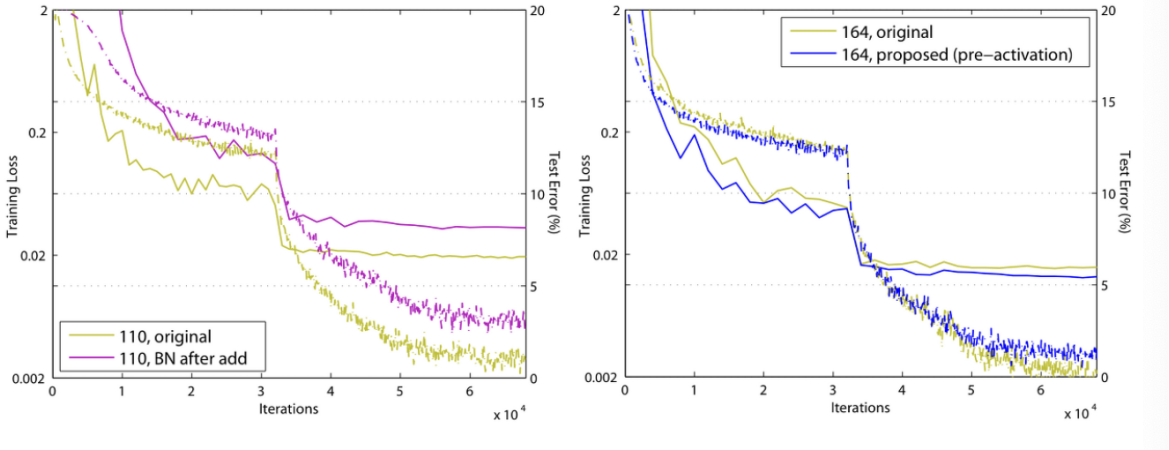
- 이 방식은 element-wise addition 이전에 ReLU를 적용시킴으로써, f 를 identity mapping으로 만들었다 그러나, residual mapping의 결과가 0에서부터 무한대까지로 제한이 된다. 즉, forward propagation 값이 단조 증가해버리는 문제가 생긴다.

3. ReLU-only pre-activation // f:identity mapping

- 이 방식은 2의 문제를 해결하기 위해 ReLU를 weight layer 이전에 적용시켰다.

4. full pre-activation // f:identity mapping

- 3의 방식을 따르면, BN의 효과를 누릴 수가 없다. 따라서 BN을 ReLU 이전에 적용시킨다.




결과를 보면, 2의 방식을 적용시킨 경우 성능이 안좋아진 반면, 4의 방식을 사용했을 때는 기존의 방식과 유사한 성능을 보였다.

| 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 |

그러나, 기존의 ResNet은 layer 깊이를 1001개로 늘렸을 때 Overfitting이 발생하는 반면 pre-activation ResNet은 Overfitting이 발생하지 않았다고 한다.

그 이유로 저자는 기존의 ResNet에서 Weight Layer의 입력이 unnormalize 상태이지만 (BN이 먼저 적용되지 않았으므로), pre-normalize 된 상태 (BN이 먼저 적용되었기 때문에) 이기 때문일 것이라고 추측하였다.

## Result

| method | augmentation | train crop | test crop | top-1 | top-5 |
|---|---|---|---|---|---|
| ResNet-152, original Residual Unit [1] | scale | 224×224 | 224×224 | 23.0 | 6.7 |
| ResNet-152, original Residual Unit [1] | scale | 224×224 | 320×320 | 21.3 | 5.5 |
| ResNet-152, **pre-act** Residual Unit | scale | 224×224 | 320×320 | 21.1 | 5.5 |
| ResNet-200, original Residual Unit [1] | scale | 224×224 | 320×320 | 21.8 | 6.0 |
| ResNet-200, **pre-act** Residual Unit | scale | 224×224 | 320×320 | **20.7** | **5.3** |
| ResNet-200, **pre-act** Residual Unit | scale+asp ratio | 224×224 | 320×320 | **20.1**† | **4.8**† |
| Inception v3 [19] | scale+asp ratio | 299×299 | 299×299 | 21.2 | 5.6 |

Pre-activation ResNet은 top-1 error를 20.1% , top-5 error를 4.8%로 기록하면서 기존의 ResNet의 성능을 앞섰고, Inception V3도 앞선 결과를 보여주었다.

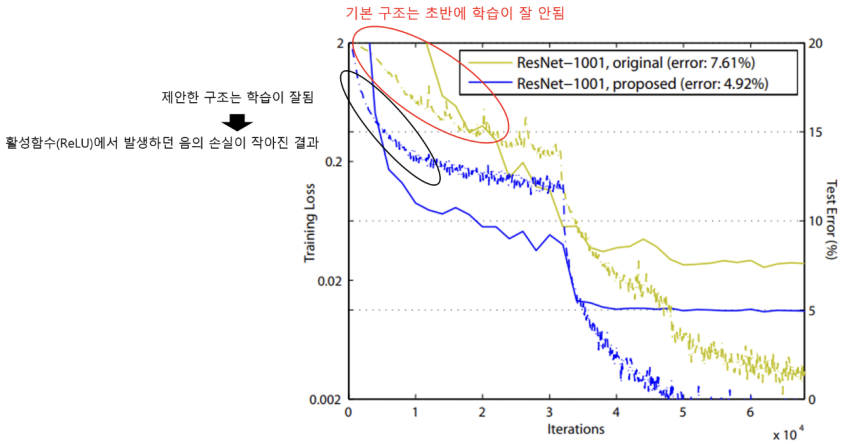
**성능분석**

post/pre activation 구조의 경향비교

Pre-activation이 주는 긍정적인 영향 중 첫 번째는 활성화 함수 f가 identity mapping이 되어 최적화하기 쉬워진 것.

ResNet의 post-activation구조에서는 ReLU로 인해 음수 신호가 모두 사라진다. 깊은 망일수록 이러한 음의 영역 신호 손실이 많이 발생하여 제일 처음 가정했던 아래 식이 만족하지 않게 된다.

이러한 현상은 ResNet-1001을 학습할 때 확인할 수 있다. post-activation 구조에서 학습할 경우 초반에 학습이 잘 이루어지지 않는 경향이 있지만 pre-activation구조에서는 처음부터 학습이 잘 이루어진다.

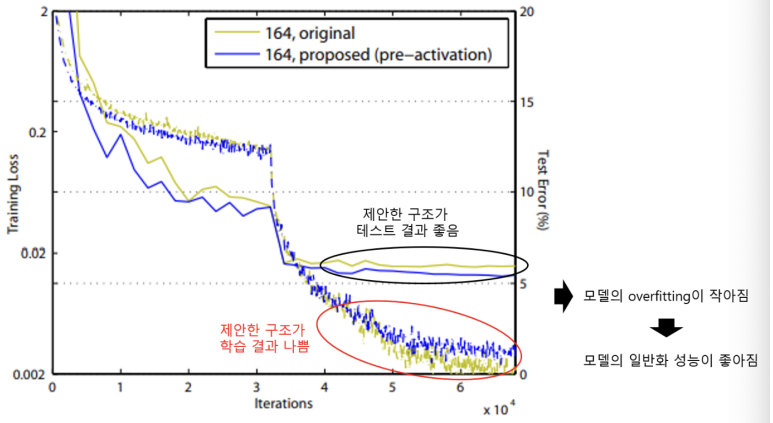



ResNet – 1001 layer

Pre-activation이 주는 긍정적인 영향 중 두 번째는 Batch Normalization의 영향으로 regularization 이 되어 일반화가 잘된다는 것.

아래 그림은 ResNet-164를 학습한 결과이다. 특이한 점은 테스트 성능은 제안된 구조가 더 좋지만, 학습 결과는 제안한 구조가 더 나쁘게 나오고 있다.

이러한 현상이 발생하는 원인은 제안하는 full pre-activation 구조는 Batch-Normalization을 통과해 정규화된 신호가 weight layer를 통과하기 때문에 일반화 성능이 올라가기 때문이다.




ResNet – 164 layer

**Table 5.** Comparisons of single-crop error on the ILSVRC 2012 validation set. All ResNets are trained using the same hyper-parameters and implementations as [1]). Our Residual Units are the full pre-activation version (Fig. 4(e)). †: code/model available at https://github.com/facebook/fb.resnet.torch/tree/master/pretrained, using scale and aspect ratio augmentation in [20].

| method | augmentation | train crop | test crop | top-1 | top-5 |
|---|---|---|---|---|---|
| ResNet-152, original Residual Unit [1] | scale | 224×224 | 224×224 | 23.0 | 6.7 |
| ResNet-152, original Residual Unit [1] | scale | 224×224 | 320×320 | 21.3 | 5.5 |
| ResNet-152, **pre-act** Residual Unit | scale | 224×224 | 320×320 | 21.1 | 5.5 |
| ResNet-200, original Residual Unit [1] | scale | 224×224 | 320×320 | 21.8 | 6.0 |
| ResNet-200, **pre-act** Residual Unit | scale | 224×224 | 320×320 | **20.7** | **5.3** |
| ResNet-200, **pre-act** Residual Unit | scale+asp ratio | 224×224 | 320×320 | **20.1**† | **4.8**† |
| Inception v3 [19] | scale+asp ratio | 299×299 | 299×299 | 21.2 | 5.6 |

다음으로 ImageNet에서 ResNet-152/200, InceptionV3를 비교한 결과이다

실험결과에서 기본 구조의 Residual Block을 사용한 ResNet-152와 ResNet-200을 비교해보면, ResNet-152가 성능이 더 좋은 것을 볼 수 있다.

특이한 점은 학습할 때 training error는 ResNet-200이 더 낮았기 때문에 논문 저자들은 Overfitting이 발생한 것으로 보았다. 하지만 일반화 성능이 좋은 pre-activation 구조의 ResNet은 200이 152보다 성능이 좋은 것을 볼 수 있다.

post-activation 구조에서는 ReLU에 의해 신호손실이 생기지만 pre-activation 구조에서는 신호 손실이 없다. 네트워크가 깊어질수록 더 많은 활성함수를 통과하기 때문에 post 구조에서 손실은 커지지만 pre-activation은 그러한 경향이 없다. 실제로 실험에서 pre-activation을 사용하면 ResNet152/200 모두 성능은 개선되지만, ResNet152 에서는 성능 개선이 미미한 것을 볼 수 있다. 하지만 ResNet200 구조에서는 상대적으로 성능이 많이 올라간 것을 볼 수 있다.

최종적으로 Pre-activation 구조를 적용하고, single crop 등의 agumentation 기법을 적용한 ResNet200이 가장 좋은 성능을 보여준다.

Pre-activation 구조는 ResNet에서 추구하던 layer가 identity-mapping이 되도록 residual block의 구조를 개선한 결과물이다.