Deep Residual Learning for Image Recognition

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun

Microsoft Research

#### Paper Link

[Deep Residual Learning for Image Recognition](https://arxiv.org/pdf/1512.03385v1.pdf)

#### Paper Explained

[[Classic] Deep Residual Learning for Image Recognition (Paper Explained)](https://www.youtube.com/watch?v=GWt6Fu05voI)

#### ResNet Implementation

[resnet.ipynb - Colaboratory](https://colab.research.google.com/github/d2l-ai/d2l-en-colab/blob/master/chapter_convolutional-modern/resnet.ipynb) [Mxnet]

[ResNet50.ipynb - Colaboratory](https://colab.research.google.com/github/yashclone999/ResNet_MODEL/blob/master/ResNet50.ipynb) [Tensorflow]

#### Summary

[7.6. Residual Networks (ResNet) — Dive into Deep Learning 0.17.5 documentation](https://d2l.ai/chapter_convolutional-modern/resnet.html)

[pending]

## Dataset

ImageNet

## Background

#### Is learning better networks as easy as stacking more layers?

ResNet addresses the degradation problem introduced in very deep neural networks. Degradation in training accuracy is often seen after a certain amount of training.

[ResNets - Deep Convolutional Models: Case Studies | Coursera](https://www.coursera.org/lecture/convolutional-neural-networks/resnets-HAhz9)

[Why ResNets Work? - Deep Convolutional Models: Case Studies | Coursera](https://www.coursera.org/lecture/convolutional-neural-networks/why-resnets-work-XAKNO)

[Why does adding layer cause degradation?](https://qr.ae/pvCEsM)

Adding more layers to a good deep model may cause degradation - reason may be that adding layers makes optimization harder. So with the existing optimization methods and initialization used, training error increases.

## Unique Features Introduced

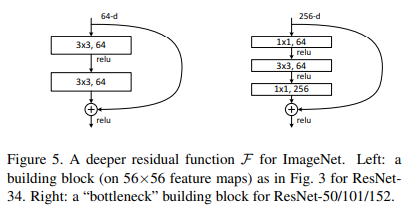
### Residual Mapping - skip connection with identity mapping or convolution mapping

Makes it easy to learn non-linear elements and also near identity elements [ weights close to zero]

***y = F(x, {Wi}) + x***

### Bottleneck Layers - the bottleneck layer is the smallest part of this network. We shrink the representation before increasing the size again.

For computational improvement, 1x1 cov is used instead of larger filter sized conv layers.



*4Deeper non-bottleneck ResNets (e.g., Fig. 5 left) also gain accuracy from increased depth (as shown on CIFAR-10), but are not as economical as the bottleneck ResNets. So the usage of bottleneck designs is mainly due to practical considerations. We further note that the degradation problem of plain nets is also witnessed for the bottleneck designs.*

## Hyperparameters Setting

*Our implementation for ImageNet follows the practice in [21, 41]. The image is resized with its shorter side randomly sampled in [256, 480] for* ***scale augmentation*** *[41]. A* ***224×224 crop is randomly sampled from an image*** *or its* ***horizontal flip****, with the per-pixel mean subtracted [21]. The standard color augmentation in [21] is used. We adopt* ***batch normalization (BN)*** *[16] right* ***after each convolution and before activation****, following [16]. We initialize the weights as in [13] and train all plain/residual nets from scratch. We use* ***SGD with a mini-batch size of 256****. The* ***learning rate starts from 0.1*** *and* ***is divided by 10 when the error plateaus****, and the models are trained for up to* ***60 × 104 iterations****. We use a* ***weight decay of 0.0001 and a momentum of 0.9****. We do not use dropout [14], following the practice in [16]. In testing, for comparison studies we adopt the standard* ***10-crop testing*** *[21]. For best results, we adopt the fully convolutional form as in [41, 13], and average the scores at multiple scales (images are resized such that the shorter side is in {224, 256, 384, 480, 640}).*

Augmentation [Best Practices for Preparing and Augmenting Image Data for CNNs](https://machinelearningmastery.com/best-practices-for-preparing-and-augmenting-image-data-for-convolutional-neural-networks/)

## Evaluation And Result

**Checkout Section 4. Experiments in Paper for more detailed experiments and results on various tasks like classification, localization, etc.**

In Table 4 we compare with the previous best single-model results. Our baseline 34-layer ResNets have achieved very competitive accuracy. Our 152-layer ResNet has a single-model top-5 validation error of 4.49%. This single-model result outperforms all previous ensemble results (Table 5). We combine six models of different depth to form an ensemble (only with two 152-layer ones at the time of submitting). This leads to 3.57% top-5 error on the test set (Table 5). This entry won 1st place in ILSVRC 2015.

