Deep Residual Learning for Image Recognition

Zhang, Liqiang

June 30, 2018

Abstract

Deeper neural networks are more difficult to train. So this paper's author present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. They explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. The author provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. The depth of representations is of central importance for many visual recognition tasks.

1. Introduction

Deep convolutional neural networks [3] have led to a series of breakthroughs for image classification. Recent evidence reveals that network depth is of crucial importance, and the leading results on the challenging ImageNet dataset all exploit "very deep" models, with a depth of sixteen to thirty. Many other nontrivial visual recognition tasks [2] have also greatly benefited from very deep models.

Driven by the significance of depth, a question arises: *Is learning better networks as easy as stacking more layers?* An obstacle to answering this question was the notorious problem of vanishing/exploding gradients, which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization and intermediate normalization layers, which enable networks with tens of layers to start converging for stochastic gradient descent (SGD) with backpropagation.

When deeper networks are able to start converging, a *degradation* problem has been exposed: with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error, as reported in [1] and thoroughly verified by the experiments. Fig. 1 shows a typical example.

In this paper, the author address the degradation problem by introducing a *deep residual learning* framework. Instead of hoping each few stacked layers directly fit a desired underlying mapping, they explicitly let these layers fit a residual mapping. Formally, denoting the desired underlying mapping as $\mathcal{H}(x)$, the author let the stacked nonlinear layers fit another mapping of $\mathcal{F}(x) := \mathcal{H}(x) - x$. The original mapping is recast into $\mathcal{F}(x)+x$. The author hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

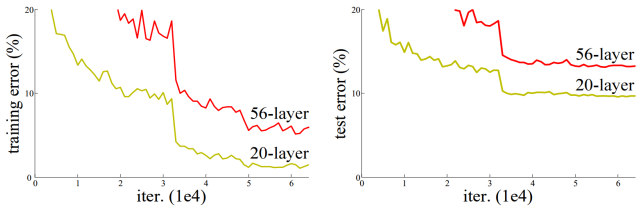

Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plai" networks. The deeper network has higher training error, and thus test error.

2. Identity Mapping by Shortcuts

The author adopt residual learning to every few stacked layers. Formally, in this paper they consider a building block defined as Eq. 1:

$$y = \mathcal{F}(x, \{W_i\}) + x. \tag{1}$$

Here x and y are the input and output vectors of the layers considered. The function $\mathcal{F}(x, \{W_i\})$ represents the residual mapping to be learned. For the example in Fig. 2 that has two layers, $\mathcal{F} = W_2\sigma(W_1x)$ in which σ denotes ReLU [4] and the biases are omitted for simplifying notations. The operation $\mathcal{F} + x$ is performed by a shortcut connection and element-wise addition.

The dimensions of x and $\mathcal{F}$ must be equal in Eq. 1. If this is not the case, it could perform a linear projection Ws by

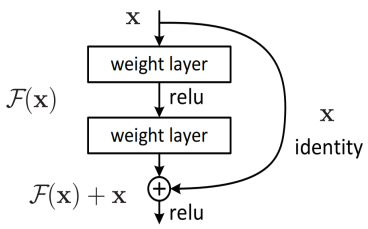

Figure 2. Residual learning: a building block.

the shortcut connections to match the dimensions as Eq. 2:

$$y = \mathcal{F}(x, \{W_i\}) + W_s x. \quad (2)$$

The form of the residual function $\mathcal{F}$ is flexible. Experiments in this paper involve a function $\mathcal{F}$ that has two or three layers, while more layers are possible. But if $\mathcal{F}$ has only a single layer, Eq. 1 is similar to a linear layer: $y = W_1 x + x$.

3. Experiments

model	top-1 err.	top-5 err.
VGG-16	28.07	9.33
GoogLeNet [5]	-	9.15
PReLU-net	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

Table 1. Error rates (%, **10-crop** testing) on ImageNet validation. VGG-16 is based on our test. ResNet-50/101/152 are of option B that only uses projections for increasing dimensions.

Table 1 shows that all three options are considerably better than the plain counterpart. B is slightly better than A. The author argue that this is because the zero-padded dimensions in A indeed have no residual learning. C is marginally better than B, and they attribute this to the extra parameters introduced by many (thirteen) projection shortcuts. But the small differences among A/B/C indicate that projection shortcuts are not essential for addressing the degradation problem.

References

[1] K. He and J. Sun. Convolutional neural networks at constrained time cost. In *CVPR*, pages 5353–5360, 2015. 1

[2] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, pages 1904–16, 2015. 1

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 1

[4] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010. 1

[5] 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, 2014. 2