Deep Residual Network Notes

Kelvin Lee

April 20, 2021

The following notes are taken from the *Deep Residual Networks for Image Recognition* by Kaiming et al. [1] and *Identity Mappings in Deep Residual Networks* by Kaiming et al. [2], which is a continuation of the previous paper.

Contents

1 Deep Residual Networks for Image Recognition

1.1 Motivation

Deeper neural networks are more difficult to train but recent evidence shows that network depth is of crucial importance. The goal is to construct an architecture that enables efficient training and higher accuracy utilizing the increased depth of the network.

1.2 Drawbacks of Increasing Layers

1.2.1 Vanishing Gradients

Can be addressed by batch normalization. See here for more details.

1.2.2 The Degradation Problem

- With network depth increasing, accuracy gets saturated and then degrades rapidly (a U-curve). Surprisingly, this is not caused by overfitting, and with more layers added to a sufficiently deep model, training error increases.
- The degradation problem suggests that the optimization solvers are not able to approximate the identity mappings of a stack of added non-linear layers (otherwise, the accuracy of the deeper network should have been at least the same as the shallower one).

1.3 Proposed Solution

Consider a shallower architecture and its deeper counterpart with more layers added onto it. There exists a solution to the deeper model by construction: the layers are copied from the learned shallower model, and the added layers are identity mapping. The existence of this constructed solution indicates that a deeper model should produce no higher training error than its shallower counterpart.

1.3.1 Residual Learning

- Let $\mathcal{H}(x)$ be the true mapping function to be learned.
- Define $\mathcal{F}(x) \doteq \mathcal{H}(x) - x$ and learn it instead of $\mathcal{H}(x)$.
- If identity mappings are optimal, the solvers may simply drive the weights of the multiple nonlinear layers toward zero to approach identity mappings.
- It is easier for the solver to find the perturbations with reference to an identity mapping than to learn the function as a new one.

1.4 Key Takeaways

- Residual Learning framework eases the training process.
- Addresses the *degradation problem* in training process.
- Leverages deeper representations of neural networks for image recognition tasks.

2 Identity Mappings in Deep Residual Networks

2.1 Overview

This paper investigates the nature of residual networks and proposes a new residual unit that allows information to flow unimpededly through the entire network by creaing a *direct* path. With this modified architecture, training of a network with layers as deep as 1000 results in further increase in accuracy.

2.2 Nature of Residual Network

2.2.1 Residual Units

Deep residual networks (ResNets) consist of many stacked *Residual Units*. Each unit has the following general form:

$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l)$$
$$\mathbf{x}_{l+1} = f(\mathbf{y}_l),$$

where $\mathbf{x}_l$ and $\mathbf{x}_{l+1}$ are input and output of the l-th layer and $\mathcal{F}$ is a residual function. Previously in [1], $h(\mathbf{x}_l) = \mathbf{x}_l$ is an identity mapping and f is a ReLU function.

Central Idea: The central idea of ResNets is to learn the additive residual function $\mathcal{F}$ with respect to $h(\mathbf{x}_l)$ using $h(\mathbf{x}_l) = \mathbf{x}_l$ by attaching a *shortcut*.

Proposition: *If both* $h(\mathbf{x}_l)$ *and* $f(\mathbf{y}_l)$ *are identity mappings*, the signal could be *directly* propagated from one unit to any other units in both forward and backward passes.

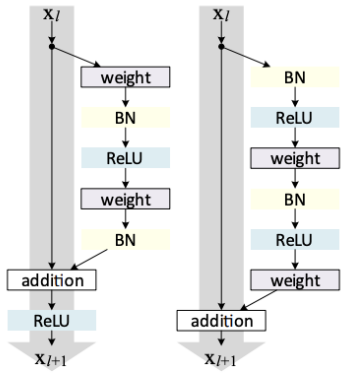

Figure 1: (a) original Residual Unit in [1]; (b) proposed Residual Unit. Grey arrows indicate the easiest paths for information propagation, corresponding to the additive term "$\mathbf{x}_l$" in Eqn.(4) (forward propagation) and the additive term "1" in Eqn.(5) (backward propagation)

.

The identity mapping $h(\mathbf{x}_l) = \mathbf{x}_l$ from [1] was found to achieve the fastest error reduction and lower training loss among all variants of h, which suggests that a *clean* information path is helpful for optimization.

To construct an identity mapping $f(\mathbf{y}_l) = \mathbf{y}_l$, view activation functions like ReLU and BN as *pre-activation* instead of *post-activation* of the weight layers. This leads to a new residual unit design shown in (Fig. 1(b)).

2.3 Analysis of Deep Residual Networks

In [1], the original Residual Unit performs the following computation:

$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l), \tag{1}$$

$$\mathbf{x}_{l+1} = f(\mathbf{y}_l). \tag{2}$$

- $\mathcal{W}_l = \{W_{l,k} \mid 1 \leq k \leq K\}$: a set of weights (and biases) from the l-th Residual Unit, where K is the number of layers in a Residual Unit.
- $\mathcal{F}$: the residual function (e.g. a stack of two convolutional layers).
- f: the operation after element-wise addition.

If f is also an identity mapping: $\mathbf{x}_{l+1} = \mathbf{y}_l$, then we have

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l). \tag{3}$$

Recursively,

$$\begin{aligned}\mathbf{x}_{l+2} &= \mathbf{x}_{l+1} + \mathcal{F}(\mathbf{x}_{l+1}, \mathcal{W}_{l+1}) \\ &= \mathbf{x}_l + \mathcal{F}(\mathbf{x}_{l+1}, \mathcal{W}_{l+1}) + \mathcal{F}(\mathbf{x}_l, \mathcal{W}_l) \\ &= \cdots\end{aligned}$$

we have

$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i=1}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i), \tag{4}$$

for deeper unit L and shallower unit l. This exhibits some nice properties:

(i) Feature $\mathbf{x}_L$ is represented as the feature $\mathbf{x}_l$ plus a residual function of form $\sum_{i=l}^{L-1} \mathcal{F}$, and hence the model is in a *residual* fashion between any units L and l.

(ii) Feature $\mathbf{x}_L$ is the *summation* of the outputs of all preceding residual functions plus $\mathbf{x}_0$. (In plain network, $\mathbf{x}_L = \prod_{i=0}^{L-1} W_i \mathbf{x}_0$, which is a *product* (ignoring BN and ReLU))

(iii) Let $\mathcal{L}$ be the loss function, then by chain rule of backpropagation, we have

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}_L} \frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}_L} \left(1 + \frac{\partial}{\partial \mathbf{x}_l} \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i)\right). \tag{5}$$

- The gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{x}_l}$ is decomposed into two additive terms: $\frac{\partial \mathcal{L}}{\partial \mathbf{x}_L}$ (which propagates information directly without concerning any weight layers), and $\frac{\partial \mathcal{L}}{\partial \mathbf{x}_L}\left(\sum_{i=l}^{L-1} \mathcal{F}\right)$ (which propagates through the weight layers).

- The first term ensures that information is directly propagated back to any shallower unit l.
- It is unlikely for the gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{x}_l}$ to vanish for mini-batch since it is not always the case that $\frac{\partial}{\partial \mathbf{x}_l} \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i) = -1$ for all samples in a mini-batch.
- Hence, the gradient would not vanish no matter how deep we go or how small the weights are.

2.3.1 Implications

The derivation of the equations above suggest that the information can be propagated from any unit to another both forward and backward as long as we have both f and h be the identity mappings. The *direct* path of flows are represented by the grey arrows from the previous figure. When the grey arrows contain no operations (except addition), the two conditions are then true and the path is then *clean*.

2.4 Summary

Identity shortcut connections and identity after-addition are essential for smooth information propagation. This enables any layer to be represented as a function of the original input. Using pre-activation ResNets by placing batch normalization and ReLU before the weights, the output of the addition becomes the output of the layer, this achieves the identity effect desired. It also helps with reducing overfitting due to BN's regularization effect. Consequently, it is possible to train residual networks as deep as 1001 layers with increasing accuracy.

References

[1] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[2] Kaiming He et al. *Identity Mappings in Deep Residual Networks*. 2016. arXiv: 1603.05027 [cs.CV].