Lab 1: VGGNet & ResNet pre-report

CHI YEONG HEO[1],
[1]School of Computer Science and Engineering, Pusan National University, Busan 46241 Republic of Korea

1. Introduction

This report provides a summary of two papers in the field of computer vision: [2], which introduces the concept of very deep convolutional networks, and [1], which presents the groundbreaking residual learning framework.

2. Very Deep Convolutional Networks for Large-Scale Image Recognition (ICLR 2015)

2.1. Overview

This paper examines the impact of convolutional network (ConvNet) depth on accuracy in large-scale image recognition tasks. The authors explore the performance improvements gained by increasing the depth of the network to 16-19 layers, while maintaining a consistent convolutional filter size of 3x3.

2.2. ConvNet Configurations

2.2.1. Architecture

The input to ConvNets is a fixed size 224x224 RGB image that is subtracted by mean RGB value of training set, from each pixel. Then the input image is passed though a stack of convolutional (conv.) layers. The size of the filters is 3x3, which is the smallest size to capture the notion of spatial information (left/right, up/down, center). The convolution stride is fixed to 1 pixel and the padding is 1 pixel so that the spatial resolution is preserved after convolution operation. The pooling operation is implemented by five max-pooling layers, each pooling layer is a 2x2 with stride 2.

Following the convolutional layers are three fully connected (FC) layers. The first two FC layers each have 4096 output channels, while the third FC layer has 1000 output channels, consistent with the classification task of the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). While the configuration of the FC layers remains constant across all networks, the configuration of the convolutional layers varies. The final layer of the network is a softmax layer, which outputs class probabilities.

Rectified Linear Unit (ReLU) activation functions are applied after every hidden layer to introduce non-linearity.

2.2.2. Configurations

The configuration of ConvNet evaluated in this paper, are outlined in Table 1. The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv<receptive field size>-<number of channels>". The ReLU activation function is not shown for brevity.

Table 1: ConvNet configurations

ConvNet Configurations					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 x 224 RGB image)					
conv-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

2.2.3. Discussion

The authors employ small 3x3 convolutional filters with a stride of 1 throughout the entire network, applying them to the input at every pixel. A stack of three convolutional layers effectively creates a receptive field of 7x7. This architectural design offers two notable advantages over a single 7x7 convolutional layer:

- **Enhanced non-linearity**: By utilizing three non-linear ReLU activations instead of one, the decision function becomes more discriminative, allowing the network to capture more complex patterns in the data.
- **Reduced parameter count**: The total number of parameters in three consecutive 3x3 convolutional layers is calculated as $3(3^2C^2) = 27C^2$, while a single 7x7 convolutional layer necessitates $7^2C^2 = 49C^2$ parameters, where C represents the number of channels.

As illustrated above, deeper architectures composed of smaller convolutional layers can provide significant advantages in terms of both performance and efficiency.

2.3. Training Configuration

Hyperparameters:

- mini-batch gradient descent with momentum 0.9
- batch size = 256
- weight decay (L_2 penalty multiplier: $5 \cdot 10^{-4}$)
- dropout for the first two FC layers (dropout ratio: 0.5)
- learning rate: start with 10^{-2}, decreased by a factor of 10 when the validation accuracy not improving

The nets required less epochs to converge due to (a) implicit regularisation imposed by greater depth and smaller conv. filter sizes; (b) pre-initialisation of certain layers.

Weight Initialisation: they initialised the first four conv. layers and the last three FC layers with layers of trained net A, which is shallow enough to be trained with random initialisation. The intermediate layers were initialised randomly with a normal distribution $\mathcal{N}(0, 10^{-2})$. The biases were initialised with zero.

2.4. Conclusion

This paper demonstrated the importance of depth in visual representations. The very deep convolutional networks (up to 19 weight layers) achieved the state-of-the-art performance on the ImageNet challenge dataset. The authors also show that their models' generalizability on various tasks and datasets, matching or outperforming more complex yet shallower image representations.

3. Deep Residual Learning for Image Recognition (CVPR 2016)

3.1. Overview

This paper introduces a residual learning framework designed to facilitate the training of very deep neural networks. The proposed framework addresses the degradation problem, wherein increasing network depth leads to saturation and even a decline in accuracy. By mitigating this issue, the residual learning approach enables the effective training of extremely deep networks, allowing them to fully exploit the benefits of increased depth. Consequently, this method achieves superior performance compared to previous architectures.

3.2. Degradation Problem

As the depth of neural networks increases, accuracy initially saturates and then rapidly degrades. This degradation is not due to overfitting, as deeper networks tend to exhibit higher training errors.

This issue is distinct from the vanishing or exploding gradient problem, which impedes convergence during the early stages of training. The latter has been addressed through techniques such as normalized initialization and the introduction of intermediate normalization layers, which allow networks to converge effectively when using stochastic gradient descent (SGD) with backpropagation.

3.3. Deep Residual Learning

3.3.1. Residual Learning

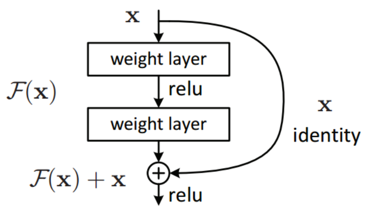
x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 1: Residual learning: a building blcok.

Residual learning framework let a stack of layers approximate a residual function, $\mathcal{F}(x) = \mathcal{H}(x) - x$ instead of $\mathcal{H}(x)$, an original function that the layers should fit. Each function is essentially equal since the layers can produce desired output with $\mathcal{F}(x) + x$. Nevertheless, learning $\mathcal{F}(x)$ is easier than learning $\mathcal{H}(x)$.

This is because the degradation problem shows that the solvers might have difficulties in learning identity mappings by multiple nonlinear layers considering that the training error of deeper model becomes greater than that of shallow model even though the added layers can be learned as identity mappings. The residual learning can easily train the layers as identity mappings by driving the weights of the layers toward zero.

3.3.2. Identity Mapping by Shortcuts

Formally, a residual building block is defined as:

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

where x and y are the input and output vectors of the layers and the function $\mathcal{F}(x, \{W_i\})$ represents the residual mapping to be learned.

In Fig. 1, $\mathcal{F} = W_2\sigma(W_1 x)$ where σ denotes ReLU and the biases are omitted for simplicity. The operation $\mathcal{F} + x$ is performed by a shortcut connection and element-wise addition. Then the second ReLU is applied after the addition. The shortcut connections introduce neither extra parameter nor computation complexity.

The shortcut connections can be applied even the dimensions of x and $\mathcal{F}$ are different by performing a linear projection W_s to match the dimensions:

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

Since the identity mapping is sufficient, W_s is only used when matching dimensions.

The from of the residual function F is flexible. F can represent not only stacks of full-connected layers but also convolutional layers. In this case, the element-wise addition is performed on two feature maps, channel by channel.

3.3.3. Network Architecture

The residual network of this paper constructed by inserting shortcut connections on the plain network. When the dimensions increase, two options are available: (A) The shortcut performs identity mapping, with extra zero entries padded for increasing dimensions, which introduces no extra parameter; (B) The projection shortcut in Eq. 2 is used to match dimensions (done by 1x1 convolutions).

The plain network is based on the VGG nets. The convolutional layers mostly have 3x3 filters and follow two simple design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer. convolutional layers have a strid of 2. The network ends with a global average polling layer and a 1000-way full-connected layer with softmax.

3.4. Conclusion

Residual learning proves to be an effective approach for mitigating the degradation problem, as it simplifies the training process by focusing on learning residual functions rather than directly approximating the target function. As demonstrated in Fig. 2, the deeper ResNet models consistently achieve lower training error compared to their shallower counterparts, whereas deeper plain networks exhibit significantly poorer performance in terms of training error. This highlights the advantages of residual learning in enabling the training of very deep networks.

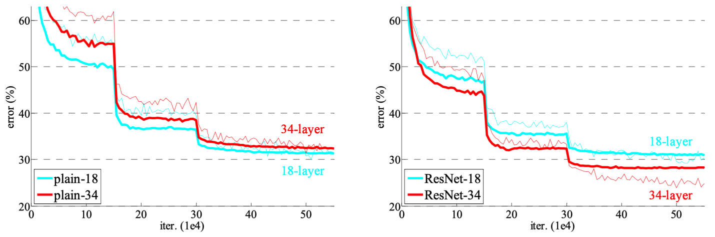

Figure 2: Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.