RV COLLEGE OF ENGINEERING

BENGALURU – 560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

"Deep Residual Learning for Image Recognition"

TECHNICAL SEMINAR (16CS82)

VIII SEMESTER

2019-2020

Submitted by

Pranay Reddy Juturu (1RV16CS064)

Under the Guidance of

Dr.Shanta Rangaswamy

RV COLLEGE OF ENGINEERING, BENGALURU - 560059

(Autonomous Institution Affiliated to VTU, Belagavi)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the **Technical Seminar Report** titled "**Deep Residual Learning for Image Recognition**" has been carried out by **Pranay Reddy Juturu (1RV16CS064),** bonafide student of R.V. College of Engineering, Bengaluru, has submitted in partial fulfillment for the **Assessment of the Course: Technical Seminar (16CS82)**during the year 2019-2020. It is certified that all corrections/suggestions indicated for the internal Assessment have been incorporated in the report.

Dr. Shanta Rangaswamy
Associate Professor,
Department of CSE,
R.V.C.E., Bengaluru –59

Dr. Ramakanth Kumar P
Head of Department,
Department of CSE,
R.V.C.E., Bengaluru–59

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends.A number of personalities ,in their own capacities have helped me in carrying out my technical seminar. I would like to take this opportunity to thank them all.

I deeply express my sincere gratitude to my guide **Dr.Shanta Rangaswamy**, Associate Professor, Department of CSE, R.V.C.E, Bengaluru,for his able guidance ,regular source of encouragement and assistance throughout this seminar

I would like to thank **Dr.Ramakanth Kumar P**, Head of Department ,Computer Science and Engineering, R.V.C.E, Bengaluru, for his valuable suggestions and expert advice.

I would like to thank **Dr.K.N. Subramanya**, Principal, R.V.C.E, Bengaluru, for his moral support towards completing my seminar.

I thank my Parents ,and all the teaching and non-teaching staff members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least ,I would like to thank my peers and friends who provided me with valuable suggestions to improve my work.

ABSTRACT

Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers - 8× deeper than VGG nets [40] but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers. The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions1, where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation..

Table of Contents

CHAPTER 1

INTRODUCTION

Deep convolutional neural networks [22, 21] have led to a series of breakthroughs for image classification [21, 49, 39]. Deep networks naturally integrate low/mid/highlevel features [49] and classifiers in an end-to-end multilayer fashion, and the "levels" of features can be enriched by the number of stacked layers (depth). Recent evidence [40, 43] reveals that network depth is of crucial importance, and the leading results [40, 43, 12, 16] on the challenging ImageNet dataset [35] all exploit "very deep" [40] models, with a depth of sixteen [40] to thirty [16]. Many other nontrivial visual recognition tasks [7, 11, 6, 32, 27] 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 [14, 1, 8], which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization [23, 8, 36, 12] and intermediate normalization layers [16], which enable networks with tens of layers to start converging for stochastic gradient descent (SGD) with backpropagation [22].

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 [10, 41] and thoroughly verified by our experiments.

CHAPTER 2

LITERATURE SURVEY

Recent advances in ML have made these techniques flexible and resilient in their applicability to various real- world scenarios, ranging from extraordinary to mundane. For instance, ML in health care has greatly improved the areas of medical imaging and computer-aided diagnosis. Ordinarily, we often use technological tools that are founded upon ML. For example, search engines extensively use ML for non-trivial tasks, such as query suggestions, spell correction, web indexing and page ranking. Evidently, as we look forward to automating more aspects of our lives, ranging from home automation to autonomous vehicles, ML techniques will become an increasingly important facet in various systems that aid in decision making, analysis, and automation.

Machine learning techniques are designed to identify and exploit hidden patterns in data for (i) describing the outcome as a grouping of data for clustering problems, (ii) predicting the outcome of future events for classification and regression problems, and (iii) evaluating the outcome of a sequence of data points for rule extraction problems. Networking problems can be formulated as one of these problems that can leverage ML. For example, a classification problem in networking can be formulated to predict the kind of security attack: Denial-of-Service (DoS),User-to-Root (U2R), Root-to-Local (R2L), or probing, given network conditions. Whereas, a regression problem can be formulated to predict of when a future failure will transpire.

Resource management in networking entails controlling the vital resources of the network, including CPU, memory, disk, switches, routers, bandwidth, AP, radio channels and its frequencies. These are leveraged collectively or independently to offer services.Admission control is an indirect approach to resource management that does not need demand prediction. The objective in admission control is to optimize the utilization of resources by monitoring and managing the resources in the network [6]. In contrast, resource allocation [7] is a decision problem that actively manages resources to maximize a long-term objective, such as revenue or resource utilization.

CHAPTER 3

DETAILED EXPLANATION OF TOPIC

3.1 Residual Learning

Let us consider H(x) as an underlying mapping to be fit by a few stacked layers (not necessarily the entire net), with x denoting the inputs to the first of these layers. If one hypothesizes that multiple nonlinear layers can asymptotically approximate complicated functions2, then it is equivalent to hypothesize that they can asymptotically approximate the residual functions, i.e., H(x) − x (assuming that the input and output are of the same dimensions). So rather than expect stacked layers to approximate H(x), we explicitly let these layers approximate a residual function F(x) :=H(x) − x. The original function thus becomes F(x)+x. Although both forms should be able to asymptotically approximate the desired functions (as hypothesized), the ease of learning might be different. This reformulation is motivated by the counterintuitive phenomena about the degradation problem (Fig. 1, left). As we discussed in the introduction, if the added layers can be constructed as identity mappings, a deeper model should have training error no greater than its shallower counterpart. The degradation problem suggests that the solvers might have difficulties in approximating identity mappings by multiple nonlinear layers. With the residual learning reformulation, if identity mappings are optimal, the solvers may simply drive the weights of the multiple nonlinear layers toward zero to approach identity mappings. In real cases, it is unlikely that identity mappings are optimal, but our reformulation may help to precondition the problem. If the optimal function is closer to an identity mapping than to a zero mapping, it should be easier for the solver to find the perturbations with reference to an identity mapping, than to learn the function as a new one. We show by experiments (Fig. 7) that the learned residual functions in general have small responses, suggesting that identity mappings provide reasonable preconditioning.

3.2. Identity Mapping by Shortcuts

We adopt residual learning to every few stacked layers. A building block is shown in Fig. 2. Formally, in this paper we consider a building block defined as:

$y = F(x,\{Wi\})+x$. (1)

Here x and y are the input and output vectors of the layers considered. The function $F(x,\{Wi\})$ represents the residual mapping to be learned. For the example in Fig. 2 that has two layers, $F = W2\sigma(W1x)$ in which σ denotes n impacted at least one of the four semen quality parameters included in the review. ReLU [29] and the biases are omitted for simplifying notations. The operation F + x is performed by a shortcut connection and element-wise addition. We adopt the second nonlinearity after the addition (i.e., σ(y), see Fig. 2). The shortcut connections in Eqn.(1) introduce neither extra parameter nor computation complexity. This is not only attractive in practice but also important in our comparisons between plain and residual networks. We can fairly compare plain/residual networks that simultaneously have the same number of parameters, depth, width, and computational cost (except for the negligible element-wise addition). The dimensions of x and F must be equal in Eqn.(1). If this is not the case (e.g., when changing the input/output channels), we can perform a linear projection Ws by the shortcut connections to match the dimensions:

$y = F(x,\{Wi\})+Wsx$. (2)

We can also use a square matrix Ws in Eqn.(1). But we will show by experiments that the identity mapping is sufficient for addressing the degradation problem and is economical, and thus Ws is only used when matching dimensions. The form of the residual function F is flexible. Experiments in this paper involve a function F that has two or three layers (Fig. 5), while more layers are possible. But if F has only a single layer, Eqn.(1) is similar to a linear layer: $y = W1x+x$, for which we have not observed advantages. We also note that although the above notations are about fully-connected layers for simplicity, they are applicable to convolutional layers. The function $F(x,\{Wi\})$ can represent multiple convolutional layers. The element-wise addition is performed on two feature maps, channel by channel

3.3. Network Architectures

We have tested various plain/residual nets, and have observed consistent phenomena. To provide instances for discussion, we describe two models for ImageNet as follows.

Plain Network. Our plain baselines are mainly inspired by the philosophy of VGG nets [40]. The convolutional layers mostly have 3×3 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. We perform downsampling directly by convolutional layers that have a stride of 2. The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax. The total number of weighted layers is 34 in. It is worth noticing that our model has fewer filters and lower complexity than VGG nets [40]. Our 34layer baseline has 3.6 billion FLOPs (multiply-adds), which is only 18% of VGG-19 (19.6 billion FLOPs).

Residual Network. Based on the above plain network, we insert shortcut connections which turn the network into its counterpart residual version. The identity shortcuts (Eqn.(1)) can be directly used when the input and output are of the same dimensions (solid line shortcuts in. When the dimensions increase, we consider two options: (A) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter; (B) The projection shortcut in Eqn.(2) is used to match dimensions (done by 1×1 convolutions). For both options, when the shortcuts go across feature maps of two sizes, they are performed with a stride of 2.

3.4. Implementation

Implementation for ImageNet follows the practice in [21, 40]. The image is resized with its shorter side randomly sampled in [256,480] for scale augmentation [40]. 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 [12] 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 [13], 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 [40, 12], and average the scores at multiple scales (images are resized such that the shorter side is in {224,256,384,480,640}).

3.5. ImageNet Classification

Plain Networks. We first evaluate 18-layer and 34-layer plain nets. The 34-layer plain net is in. The 18-layer plain net is of a similar form. See Table 1 for detailed architectures. The results in Table 2 show that the deeper 34-layer plain net has higher validation error than the shallower 18-layer plain net. To reveal the reasons) we compare their training/validation errors during the training procedure.

Deeper Bottleneck Architectures. Next we describe our deeper nets for ImageNet. Because of concerns on the training time that we can afford, we modify the building block as a bottleneck design4. For each residual function F, we use a stack of 3 layers instead of 2. The three layers are 1×1, 3×3, and 1×1 convolutions, where the 1×1 layers are responsible for reducing and then increasing (restoring) dimensions, leaving the 3×3 layer a bottleneck with smaller input/output dimensions. Where both designs have similar time complexity. The parameter-free identity shortcuts are particularly important for the bottleneck architectures. If the identity shortcut in is replaced with projection, one can show that the time complexity and model size are doubled, as the shortcut is connected to the two high-dimensional ends. So identity shortcuts lead to more efficient models for the bottleneck designs.

50-layer ResNet: We replace each 2-layer block in the 34-layer net with this 3-layer bottleneck block, resulting in a 50-layer ResNet (Table 1). We use option B for increasing dimensions. This model has 3.8 billion FLOPs.

101-layer and 152-layer ResNets: We construct 101layer and 152-layer ResNets by using more 3-layer blocks (Table 1). Remarkably, although the depth is significantly increased, the 152-layer ResNet (11.3 billion FLOPs) still has lower complexity than VGG-16/19 nets (15.3/19.6 billion FLOPs). The 50/101/152-layer ResNets are more accurate than the 34-layer ones by considerable margins. We do not observe the degradation problem and thus enjoy significant accuracy gains from considerably increased depth. The benefits of depth are witnessed for all evaluation metrics.

Comparisons with State-of-the-art Methods. 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. 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 . This entry won the 1st place in ILSVRC 2015

CHAPTER 4

CONCLUSION

The depth of neural networks is absolutely important to obtaining better performance. When neural network layers are very deep, they exhibit higher training and validation loss due to information loss leading to vanishing gradients. To guarantee that deeper networks would always yield better accuracy than shallow networks, the authors of Resnet proposed the use of residual blocks, bringing in information from the past to compensate for information loss. This technique enables the training of very deep networks resulting in state of the art accuracy on standard benchmarks.

To reduce computation cost due to very deep layers, the authors use bottleneck layers with 1 x 1 convolutions that reduce the number of feature maps of the input.

REFERENCES

[1] R. Girshick. Fast R-CNN. In ICCV, 2015.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[3] K. He and J. Sun. Convolutional neural networks at constrained time cost. In CVPR, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, 2014.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In ICCV, 2015.

[6] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

[7] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[8] R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. arXiv:1505.00387, 2015.

[9] T. Raiko, H. Valpola, and Y. LeCun. Deep learning made easier by linear transformations in perceptrons. In AISTATS, 2012.

[10] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015.