
Investigating the Optimization of Convolutional Networks-Vanishing Gradient and degradation problem

Abstract

Vanishing gradient issue and degradation problem are identified as the optimization problems that VGG38 training encountered. After reviewing related literature, we implemented the Batch Normalization mechanism at first, which effectively enabled broken VGG38 to begin converging. However, the achieved training accuracy—0.572 still worse than much shallower VGG08 and exposed a degradation problem. By integrating Residual Learning Framework further, the training accuracy rapidly increased to 0.862 and easily beat the baseline. Finally, by handling overfitting further, we achieved a stable test accuracy that 0.651 ± 0.0048 .

1. Introduction

1.1. Motivation-A Counterintuitive Phenomenon

Due to the increasing requirements in visual object recognition, Convolutional Neural Networks (CNNs) have gained great attention and recent works (Szegedy et al., 2014) reveal that network depth is of crucial significance for handling complex tasks. Theoretically, as the network depth going up, CNNs are supposed to be able to gain benefits from the abstraction power, enriched features provided by extra layers, at least it should not perform worse when the shallower model nested into it (He et al., 2016). However, as shown in Figure 1, the deeper VGG38 unexpectedly performs worse than the much shallower VGG08. Surprisingly, such phenomenon is not caused by overfitting, for it is also observed on training set. Motivated by this counterintuitive phenomenon, this paper aims to explore the optimization problems behind it.

1.2. Overview

In the first place, this paper recognizes the vanishing gradient problem and degradation problem as reasons that lead to the counterintuitive phenomenon. Vanishing gradient is a well-known problem in deep networks optimization, which jeopardizes the convergence from beginning. In this paper, vanishing gradient problem was identified by visualizing the mean absolute gradient of VGG38's each layer. However, degradation problem, which refers to the unexpected lower training accuracy caused by optimization difficulty in deep network, is relatively hard to identify for it usually hides behind the vanishing gradient problem. **Therefore, in this paper, the vanishing gradient problem was solved**

at first. Then the degradation problem can be exposed and identified, by comparing the training performance of VGG08 and VGG38 (with repaired gradient).

Secondly, to achieve a better understanding and find the possible solutions for the identified problems, three literatures are reviewed (Ioffe & Szegedy, 2015), (He et al., 2016), (Huang et al., 2017). Except Dense connection, both BN and RL were adopted to solve the two identified problems respectively. BN was originally proposed to solve “internal covariate shift” problem, but by normalizing the activation it can deal with vanishing and gradient problem, for it prevents the small variation of a layer’s weights from being magnified too much (gradient exploding) or decreased to zero (gradient vanishing). Residual learning was developed for solving degradation problem exactly, by fitting optimization-friendly residual mapping, and it can also address vanishing gradient problem for the short connections can facilitate the flow of information through the network.

Adopting residual learning technique can solve the two problems at once, but the degradation problem hidden behind the vanishing gradient issue will never be exposed. Therefore, BN was applied at first, and then RL was introduced to address the remaining degradation problem. The experiments were done on the CIFAR100 dataset, which consists of 60,000 32×32 color images with 600 images contained in each of 100 classes. By adopting BN, the gradient of VGG38 network was repaired, but the semi-healthy VGG38 still struggling to gain accuracy from its depth and can not beat shallower model, which implies the appearance of degradation problem. Finally, a totally healthy VGG38 was established by further introducing RL technique, then the capacity of VGG38 was fully recovered and can easily beat VGG08 and solely BN-based VGG38 in terms of training accuracy, validation accuracy and convergence speed. After solving the overfitting problem, the test accuracy of the final model reported on multiple seed is 0.651 ± 0.0048 .

2. Identifying training problems of a deep CNN

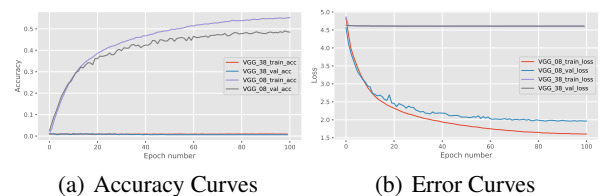


Figure 1. Performance Comparison:VGG08 and Broken VGG38

Internal Covariate Shift, which defined as the changing activation distribution caused by parameters update of preceding layers, cause a problem for each layer has to keep adapting the changing distribution of its activation. To mitigate this problem, Batch Normalization was proposed (Ioffe & Szegedy, 2015). Firstly, inspired by the idea of whitening, the study intended to do normalization on each layer’s inputs. Then for reducing unaffordable computational complexity, two simplifications were made, that normalize each scalar feature independently rather than jointly and calculate its mean and variance on a mini-batch instead of on the

whole training set. Besides, two learnable parameters were introduced for recovering the layer's representation ability affected by normalization. The study indicated that by using Batch Normalization the training of deep network can be greatly accelerated and can afford higher learning rate as well as need less weight penalty. Besides, it also benefit for solving vanishing gradient problem for the optimizer is less likely stuck in the saturated region of nonlinearity, and gradients are less depend on the scale of the parameters or of their initial values.

Experimentally, the benefits of Batch Normalization are obvious, but the reasons about why BN can promote training stay controversial. Recently, some works reveal that instead of mitigating Internal Covariate Shift, BN smoothed the objective function (Santurkar et al., 2019).

3.2. Deep Residual Learning for Image Recognition

Motivated by the counterintuitive phenomenon, that deeper network unexpectedly perform worse than the corresponding shallower network, the study conducted by (He et al., 2016) aims to solve this degradation problem. To this end, Residual Learning Framework was proposed, where each layer learns the residual mapping rather than unreferenced original mapping, for it was assumed easier to optimize. The authors stated residual mapping can only improve the capacity of layers for it can be degraded to original mapping by pushing residual to zero. Residual mapping was implemented by adding "shortcut connection", which will not add extra parameters when identity connection is added. Besides, they also provided three strategies to enable shortcut connection match the dimension, and the comparisons was well discussed. During the study, a series of experiments were presented, where residual learning framework successfully enabled deeper network gain accuracy from its depth. However, when network's depth exceeds 1000 layers the test accuracy decreased, and the authors believed this phenomenon was caused by overfitting.

This study fully discussed the details of experimental design and clearly explained the motivation behind their choices. However, it failed to identify the reason caused the degradation problem and it's beneficial effect for handling gradient vanishing problem are rarely explained. **It is worth mentioning that**, Batch Normalization were applied throughout the study, not only because vanishing gradient problem of baseline need to be fixed at first, but also for taking control experiments, which inspired our works.

3.3. Dense connected network

Inspired by the idea of "short connection", which can promote the information flow through the network, DesNet (Huang et al., 2017) was developed for enabling maximum information flow through the network. Similar but different with residual learning framework, DenseNet's each layer is connected to every subsequent layer. Different with ResNets where the features are summed together, DesNet choose to concatenate them. The authors stated that the

inputs concatenation and the dense connection can create a map of collective knowledge, which can be accessed and reused by any following layers. One benefit of this is, there is no need for relearning redundant feature-map, and thus the DesNet need less parameter and less prone to overfit the data. Except the benefit of fewer parameter, better feature reuse, DesNet improves the conduction of information and gradient, which can solve gradient vanishing problem and promote the training. Finally, to evaluate the practicability of DesNet, DesNet were applied on four competitive object recognition tasks and achieve a great performance on all of them.

Comparisons: All the three approaches can handle vanishing gradient problems, but in two different ways. BN did it by decoupling the dependency between gradients and the scale of parameters, but ResNet and DesNet did it by adding short connections which can encourage the propagation of information and error signal. Only ResNet and DesNet can handle the degradation problem for the similar idea they started from. The key difference between ResNet and DesNet is the way they handle the identity mappings. ResNet sum them together $\mathbf{x}_\ell = H_\ell(\mathbf{x}_{\ell-1}) + \mathbf{x}_{\ell-1}$, but DesNet concatenate them $\mathbf{x}_\ell = H_\ell([\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{\ell-1}])$. The concatenation maximally reserved the information and provide a collective knowledge map for the subsequent layers to access. DesNet also less likely overfit the data which provide a solution for the overfitting problem occurred in ResNet. However, compared with BN and Residual Learning which can be easily integrated into existed network, DenseNet is less combinable for the relatively complicated architecture.

4. Solution Overview

4.1. Batch Normalization

As seen in section 3, Batch Normalization is beneficial for solving vanishing problem, and it can be easily integrated into the existed network. Besides, compared with the alternative that carefully initializing weights (Pascanu et al., 2013), adopting BN allow us concern less of the initialization and training procedures. Therefore, Batch Normalization is implemented at first to solve the gradient vanishing problem. **To be more specific**, before the activation going into ReLu, BN transformation was applied on each dimension of the layer's activation independently. Formulas of BN transformation are:

$$\begin{aligned} \mu_{\mathcal{B}} &\leftarrow \frac{1}{m} \sum_{i=1}^m x_i & \sigma_{\mathcal{B}}^2 &\leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \\ \hat{x}_i &\leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} & y_i &\leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma\beta}(x_i) \end{aligned} \quad (2)$$

Where $\mathcal{B} = \{x_{1...m}\}$ is the mini-batch that $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ calculated on, and $\{x_{1...m}\}$ is the activations that BN applied on. γ and β are two parameters to be learned, which enable activation go into any part of the non-linear function. ϵ is a constant added for guarantee numerical stability. The

Figure 4. Gradient Flow of BN-based VGG38

Note that for taking control experiment, BN is also applied on VGG08, and it will be unfair if just allowing VGG38 take advantage from a useful technique.

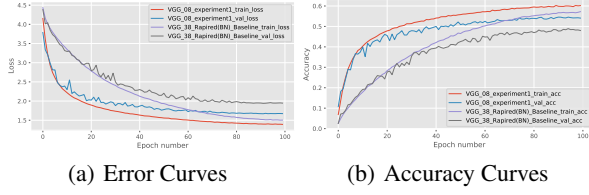


Figure 5. Comparison of BN applied VGG38 and VGG08

From Figure 3, 4 we can found that the gradient signal successfully flowed back to the input layer and VGG38 begin converging. However, the gradient-healthy VGG38 (0.572 training accuracy and 0.48 validation) still cannot beat the much shallower VGG08 (0.602 training accuracy, 0.541 validation accuracy) on training set, and the relatively stable tails of curves imply increasing epoch number is less likely to improve performance. *But what if this simply because the advantages of BN has not been fully exploited?*

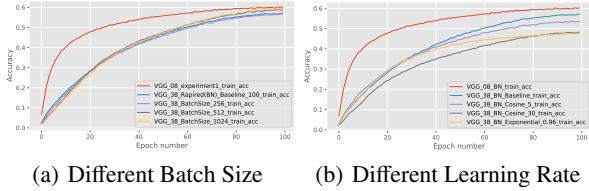


Figure 6. Training Accuracy against Parameter Setting

Since we noticed the simplification made in the original article, that calculate means and variances on mini-batch rather than on the whole training set, we first increased the batch size for larger batch size can provide better approximation and thus have better performance. According to Figure 6 (a), the training accuracy did increase with the batch size, but **the highest training accuracy (0.595) achieved at 1024 batch size still worse than the baseline.** We did not keep increasing batch size for the memory limit of CPU and the tiny increase of VGG38's training accuracy is not worth to explore further. **Interesting to mention, the running time decreased with the increase of batch size.** Secondly, as the original paper suggested, we increased the initial learning rate to 5 and 30 times of the defaulted $1e^{-3}$. However, as Figure 6(b) shown, the increased learning rate can only decrease the training performance. Besides, we also changed the learning schedule to exponential learning schedule (0.96 decay rate) to in accordance with the original paper. Still no clues show we should explore this aspect further. We didn't change the weight decay coefficient for BN itself can mitigate overfitting, and there is no sign of overfitting but underfitting. **Till now we can sure vanishing gradient is not the only problem that prevent VGG38 gain benefit from its depth, and the degradation problem is explicitly exposed.**

5.2. Solving Remaining Degradation Problem

Similar with the experimental procedures of (He et al., 2016), RL was implemented on the basis of the Batch Nor-

malization. Without Batch Normalization, the semi-healthy VGG38 baseline cannot begin converging and thus cannot be compared with our experiment results. Following the design in original paper, a identity mapping from the second previous layer was added on the batch normalized output of current layer before going to leaky Relu function. Besides, all the setting still keep default, for we only care whether degradation problem can be solved in this subsection. The training performance comparisons of semi-healthy VGG38 (only use BN) and healthy VGG38 (BN+ResNet) is shown below:

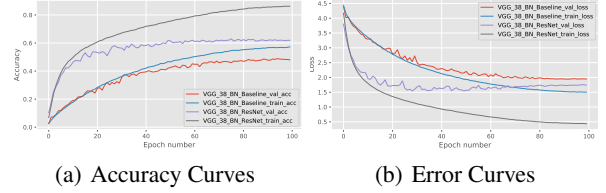


Figure 7. Performances of Semi-healthy and Healthy VGG38

NETWORK	TRA _{acc}	TRA _{err}	VAL _{acc}	VAL _{err}
VGG08-BN	0.602	1.392	0.541	1.673
VGG38-BN	0.572	1.503	0.48	1.947
VGG38-BN+RES	0.862	0.438	0.619	1.739

Table 1. Accuracies Errors of semi-healthy and healthy VGG38

As shown in Figure 7(a) and Table 1, with the help of Residual Learning Framework the healthy VGG38 easily exceeded both BN-applied VGG08 and the semi-healthy VGG38 by about 29 percent training accuracy. For exploring how Residual Learning affected gradient signal, the gradient flow of healthy-VGG38 was plotted.

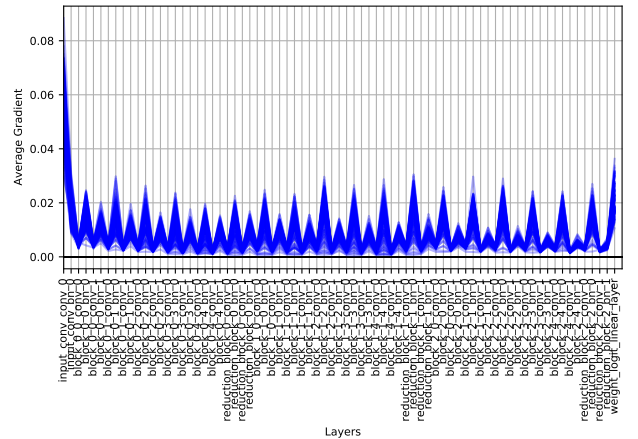


Figure 8. Gradient Flow of BN Assembled VGG38

5.3. Solving Overfitting and Testing Stability

Till now the vanishing gradient problem and degradation problem have been well settled. This experiment aims to solve the overfitting problem exposed in Figure 7(b), for achieving a better and more stable performance. Instead of using dropout we increased the weight decay coefficient, because dropout keep changing the network architecture for each batch and thus a little bit risky. The coefficient was adjusted from $1e^{-4}$, for the original paper (He et al.,

2016) adopted this value on VGG network and gained great performances. The searching stopped at $1e-2$ for the validation accuracy curve not stable anymore.

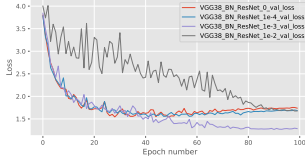


Figure 9. Validation Error

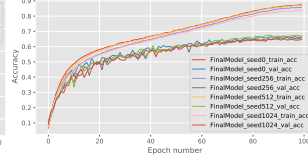


Figure 10. Test Stability

As shown in Figure 9, when coefficient equal to $1e-3$ the overfitting problem disappeared, thus we adopted this coefficient and run our final model on multiple seed to test its stability (results shown in Figure 10 and Table 2).

SEED	TRA _{acc}	TRA _{err}	VAL _{acc}	VAL _{err}	TEST _{err}	TEST _{err}
0	0.874	0.435	0.663	1.286	0.655	1.312
256	0.859	0.487	0.648	1.343	0.646	1.360
512	0.871	0.451	0.670	1.273	0.657	1.321
1024	0.838	0.552	0.658	1.300	0.647	1.316

Table 2. Accuracies Errors of semi-healthy and healthy VGG38

According to Table 2, the final test accuracy is reported as 0.651 ± 0.0048 (*mean* \pm *std*), and loss is 1.327 ± 0.0191 . We did not do fine-grained searching and other hyperparameters searching further, for our purpose is not provide a model with best performance, but solving the identified problem and let model achieve a stable performance.

6. Discussion

Firstly, compared with Figure 2, the gradients shown in Figure 3 successfully flowed back to the input layer. However, the peaky shape of the gradient flow is unexpected in advance. Following the notation in section 4.1, where BN represents the BN transformation which applied on each dimension of $A^{(n)} = W^{(n)}h^{(n-1)}$ independently, $A^{(n)}$ and $h^{(n-1)} = g(BN(A^{(n-1)}))$ are current layer's activation and input respectively. According the definition of BN , $BN(A^{(n)}) = BN((aA^{(n)}))$, where a is a scalar. Then $\frac{\partial BN(aA^{(n)})}{\partial A^{(n)}} = \frac{1}{a} \cdot \frac{\partial BN(A^{(n)})}{\partial A^{(n)}}$, which means smaller weights will lead to smaller a and then gradients of current layer's activation will be amplified when gradient signal flow to it. **The activation's gradient is just the gradients shown on 'bn' layers**, so the gradient flow show a upward trend on 'bn' layers. However, when the gradient signal flow to the $W^{(n)}$, a tiny value $h^{(n-1)}$ will be multiplied on it, thus the gradient flow show a downward trend on CONV layers. The healthy gradient flow and the performance curves shown in Figure 4, indicates BN enabled broken VGG38 to converge and solved the vanishing gradient problem effectively. However, its training accuracy 0.572 shows a underfitting problem and obviously cannot match its expected fitting capacity. Besides the experimental results given in Figure 5 shows that reasonably changing hypermeters still cannot enable semi-healthy VGG38 gain accuracy from its depth and exceed baseline in terms of training accuracy, which make the degradation problem more explicit. Run time decrease with the increase of batch size may because larger batch

size will lead to less mean and variation calculation. Meanwhile, contrary to the original literature (Ioffe & Szegedy, 2015), increasing learning rate can only make things worse. This may partly due to the experiments were run on different datasets for different tasks, but more likely because the degradation problem prevented the network from gaining benefit from BN further.

The huge training accuracy improvement (29% shown in Figure 6(a) and Table 1 indicates Residual Learning Framework empowered the semi-healthy VGG38 to reap benefits from the extra layers and beat the baseline easily. Besides, the overfitting phenomenon shown in Figure 6(b) also indicates the fitting ability of semi-healthy VGG38 has been recovered successfully. Therefore the **Residual Learning Framework successfully handled the degradation problem**. However, the experiments conducted by (He et al., 2016) on the CIFAR-10 dataset, shows the overfitting problem occurs only when the depth of network exceeds 110 layers. This may because our training images for each class is 10 times smaller then their, for we classified the training images into 100 classes rather than their 10 classes, thus our VGG38 may unnecessary deep for the smaller dataset.

There is an interesting phenomenon discovered in Figure 8. Compared with the gradient flow of semi-healthy VGG38, after applying Residual Learning Framework further the gradient of 'bn1' where identity mapping was added has reduced. This because 'bn1' actually learnt the residual mapping. Compared with learning underlying mapping directly, learning residual mapping need weights of this layer change less for it has a identity mapping as a reference. Consequently, the layer will have smaller gradient for its weights contribute less to the output. Besides, as the depth increasing the reference should be more and more accurate for the increasing abstraction power provided by preceding stacked layers, and thus the weights of deeper layer will change less. Indeed we can found this trend on the Figure, where the 'bn1' near the input has larger gradient and the 'bn1' near the output has smaller gradient.

7. Conclusions

In this report we explored the optimization problem of deep CNN training that vanishing gradient issue and degradation problem. After applying Batch Normalization and integrating Residual Learning Framework further, we successfully addressed these problems. Finally, after solving overfitting, We achieved a reasonable training accuracy that 0.861 ± 0.0142 and stable test accuracy that 0.651 ± 0.0048 . During this procedure we not only have a deeper understanding of these optimization problem, we also learned how to apply the learned knowledge into practice. Besides, we also learned it is better to start from some simple baselines and do experiment incrementally, for it may expose more hidden problems, such as the degradation problem found in this report. In the future we would like to explore the causes of the degradation problem and the real principle of Batch Normalization for it stay controversial.

References

- Balduzzi, David, Frean, Marcus, Leary, Lennox, Lewis, J. P., Ma, Kurt Wan-Duo, and McWilliams, Brian. The shattered gradients problem: If resnets are the answer, then what is the question? *CoRR*, abs/1702.08591, 2017. URL <http://arxiv.org/abs/1702.08591>.
- He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. IEEE, 2016. ISBN 9781467388511.
- Huang, Gao, Liu, Zhuang, van der Maaten, Laurens, and Weinberger, Kilian Q. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. 2015.
- Krizhevsky, Alex et al. Learning multiple layers of features from tiny images. 2009.
- Pascanu, Razvan, Mikolov, Tomas, and Bengio, Yoshua. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pp. 1310–1318, 2013.
- Santurkar, Shibani, Tsipras, Dimitris, Ilyas, Andrew, and Madry, Aleksander. How does batch normalization help optimization?, 2019.
- Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. 2014.