Paul Aggarwal

Big Transfer

QUESTION 2:

All BiT models use a vanilla ResNet-v2 architecture (ref. 201603.05027.pdf), except that they replace all Batch Normalization layers with Group Normalization and use Weight Standardization in all convolutional layers. Compared to ResNet-v1 architecture stacks of 2 x (3 x 3) Conv2D-BN-ReLU, the Resnet-v2 has stacks of (1 x 1) – (3 x 3) – (1 × 1) BN-ReLU-Conv2D where Batch normalizations and ReLU activations come before two dimensional convolution layers, not after the convolutional layers. Also, in this case with BiT, BN was replaced with GN+WS. ResNets which have residual blocks (stacks) can overcome vanishing gradients in deep networks.

They train ResNet-152 architectures in all datasets, with every hidden layer widened by a factor of four (ResNet152x4). They study different model sizes and the coupling with dataset size. They train all of the models upstream using SGD with momentum. They use an initial learning rate of 0.03, and momentum 0.9. During image preprocessing stage they use image cropping technique from, and random horizontal mirroring followed by 224 × 224 image resize. They train both BiT-S and BiT-M for 90 epochs and decay the learning rate by a factor of 10 at 30, 60 and 80 epochs. For BiT-L, they train for 40 epochs and decay the learning rate after 10, 23, 30 and 37 epochs. They use a global batch size of 4096 and train on a Cloud TPUv3- 512 [24], resulting in 8 images per chip. They use linear learning rate warm-up for 5000 optimization steps and multiply the learning rate by batch size 256. During pre-training they use a weight decay of 0.0001, but do not use any weight decay during transfer.

To attain a low per-task adaptation cost, they did not perform any hyperparameter sweeps downstream. Instead, they presented BiT-HyperRule, a heuristic to determine all hyperparameters for fine-tuning. Most hyperparameters are fixed across all datasets, but schedule, resolution, and usage of MixUp depend on the tasks image resolution and training set size. For all tasks, they use SGD with an initial learning rate of 0.003, momentum 0.9, and batch size 512. They resize input images with area smaller than 96 × 96 pixels to 160 × 160 pixels, and then take a random crop of 128 × 128 pixels. They resize larger images to 448 × 448 and take a 384 × 384-sized crop.1 They apply random crops and horizontal flips for all tasks, except those for which cropping or flipping destroys the label semantics. For schedule length, they define three scale regimes based on the number of examples: they call small tasks those with fewer than 20 k labeled examples, medium those with fewer than 500 k, and any larger dataset is a large task. They fine-tune BiT for 500 steps on small tasks, for 10k steps on medium tasks, and for 20k steps on large tasks. During fine-tuning, they decay the learning rate by a factor of 10 at 30%, 60% and 90% of the training steps. Finally, they use MixUp, with $\alpha = 0.1$, for medium and large tasks.

The ResNet larger models have a high memory requirement for any single accelerator chip, which necessitates small per-device batch sizes. However, BN performs worse when the number of images on each accelerator is too low. An alternative strategy is to accumulate BN statistics across all of the accelerators. However, this has two major drawbacks. First, computing BN statistics across large batches has been shown to harm generalization. Second, using global BN requires many aggregations across accelerators which incurs significant latency. It was investigated Group Normalization (GN) and Weight Standardization (WS) as alternatives to BN. It was tested large batch training using 128 accelerator chips

and a batch size of 4096. It was found that GN alone does not scale to large batches; they observe a performance drop of 5.4% on ILSVRC-2012 top-1 accuracy compared to using BN in a ResNet-50x1. The addition of WS enables GN to scale to such large batches, even outperforming BN. They are not only interested in upstream performance, but also how models trained with GN and WS transfer. They thus transferred models with different combinations of BN, GN, and WS pre-trained on ILSVRC-2012 to the 19 tasks defined by VTAB. The results in Table 5 indicate that the GN/WS combination transfers better than BN, so they use GN/WS in all BiT models.

ResNets are fully convolutional by design. Resnets obtain good results via a simple but essential concept — going deeper. Resnets do not specially tailor the network width or filter sizes, nor use regularization techniques (such as dropout) which are very effective for these small datasets.

Experiments support the analysis in Eqn.(5, ref. 201603.05027.pdf) and Eqn.(8, ref. 201603.05027.pdf), both being derived under the assumption that the after-addition activation f is the identity mapping. But in the experiments f is ReLU as designed, so Eqn.(5, ref. 201603.05027.pdf) and (8, ref. 201603.05027.pdf) are approximate in the experiments. Next, they investigate the impact of f. They want to make f an identity mapping, which is done by re-arranging the activation functions (ReLU and/or BN). The original Residual Unit has a shape in Fig. 4(a) (ref. 201603.05027.pdf) — BN is used after each weight layer, and ReLU is adopted after BN except that the last ReLU in a Residual Unit is after elementwise addition (f = ReLU). Fig. 4(b-e) (ref. 201603.05027.pdf) show the alternatives they investigated, explained as following. (ref. 201603.05027.pdf)

The experiment with ResNet-110 and a 164-layer Bottleneck (Resnet-V1) architecture (denoted as ResNet-164). Changing bottleneck Residual Unit to a 1×1 layer for reducing dimension, a 3×3 layer, and a 1×1 layer for restoring dimension. As designed, this becomes Resnet-V2 architecture, its computational complexity is similar to the two-3×3 Residual Unit. The baseline ResNet164 has a competitive result of 5.93% on CIFAR-10.

BN after addition. Before turning f into an identity mapping, they go the opposite way by adopting BN after addition (Fig. 4(b) (ref. 201603.05027.pdf)). In this case f involves BN and ReLU. The results become considerably worse than the baseline (Table 2). Unlike the original design, now the BN layer alters the signal that passes through the shortcut and impedes information propagation, as reflected by the difficulties on reducing training loss at the beginning of training (Fib. 6 left, ref. 201603.05027.pdf). ReLU before addition. A naive choice of making f into an identity mapping is to move the ReLU before addition (Fig. 4(c), ref. 201603.05027.pdf). However, this leads to a non-negative output from the transform F, while intuitively a “residual” function should take values in (−∞, +∞). As a result, the forward propagated signal is monotonically increasing. This may impact the representational ability, and the result is worse (7.84%, Table 2, ref. 201603.05027.pdf) than the baseline. They expect to have a residual function taking values in (−∞, +∞). This condition is satisfied by other Residual Units including the following ones. (ref. 201603.05027.pdf)

It is easy to see that Eqn.(9, ref. 201603.05027.pdf) is similar to Eqn.(4, ref. 201603.05027.pdf), and can enable a backward formulation similar to Eqn.(5, ref. 201603.05027.pdf). For this new Residual Unit as in Eqn.(9, ref. 201603.05027.pdf), the new after-addition activation becomes an identity mapping. This design means that if a new after-addition activation $\hat{f}$ is asymmetrically adopted, it is equivalent to recasting $\hat{f}$ as the pre-activation of the next Residual Unit. Using asymmetric after-addition activation is equivalent to constructing a pre-activation Residual Unit. (ref. 201603.05027.pdf)

The distinction between post-activation/pre-activation (respectively on ResNet-v1/Resnet-v2) is caused by the presence of the element-wise addition. For a plain network that has N layers, there are N − 1 activations (BN/ReLU), and it does not matter whether they think of them as post- or pre-activations. But for branched layers merged by addition, the position of activation matters.

When BN and ReLU are both used as pre-activation, the results are improved by healthy margins. Various architectures: (i) ResNet-110, (ii) ResNet-164, (iii) a 110-layer ResNet architecture in which each shortcut skips only 1 layer (i.e., a Residual Unit has only 1 layer), denoted as "ResNet-110(1layer)", and (iv) a 1001-layer bottleneck architecture that has 333 Residual Units (111 on each feature map size), denoted as "ResNet-1001". These "pre-activation" models are consistently better than the baseline counterparts.

Ease of optimization effect is particularly obvious when training the 1001-layer ResNet. Using the original design, the training error is reduced very slowly at the beginning of training. For f = ReLU, the signal is impacted if it is negative, and when there are many Residual Units, this effect becomes prominent and Eqn.(3, ref. 201603.05027.pdf) (so Eqn.(5), ref. 201603.05027.pdf) is not a good approximation. On the other hand, when f is an identity mapping, the signal can be propagated directly between any two units. Our 1001-layer network reduces the training loss very quickly It also achieves the lowest loss among all models they investigated, suggesting the success of optimization. They also find that the impact of ReLU is not severe when the ResNet has fewer layers. The training curve seems to suffer a little bit at the beginning of training but goes into a healthy status soon. By monitoring the responses they observe that this is because after some training, the weights are adjusted into a status such that yl in Eqn.(1, ref. 201603.05027.pdf) is more frequently above zero and f does not truncate it (xl is always non-negative due to the previous ReLU, so yl is below zero only when the magnitude of F is very negative). The truncation, however, is more frequent when there are 1000 layers. (ref. 201603.05027.pdf)

Another impact of using the proposed pre-activation unit (ResNet-v2) is on regularization (reducing overfitting). The pre-activation version reaches slightly higher training loss at convergence but produces lower test error. This phenomenon is observed on ResNet-110, ResNet-110(1-layer), and ResNet-164 on both CIFAR-10 and 100. This is presumably caused by BN's regularization effect. In the original Residual Unit, although the BN normalizes the signal, this is soon added to the shortcut and thus the merged signal is not normalized (ResNet-V1 version). This unnormalized signal is then used as the input of the next weight layer. On the contrary, in our pre-activation version (ResNet-v2), the inputs to all weight layers have been normalized.

In conclusion, "pre-activation" models (ResNet-v2) are consistently better than the baseline counterparts (ResNet-v1) which are "post-activation" models. They note that they do not specially tailor the network width or filter sizes, nor use regularization techniques (such as dropout) which are very effective for these small datasets. But now, they are able to achieve good results on very large datasets by replacing the BN with GN+WS and upgrade the Resnet-v1 architecture with Resnet-v2 and simply using GN+WS, RELU before the two dimensional convolution layers which uses stacks of (1 x 1)-(3 x 3)-(1 x 1) [GN+WS]-ReLU-Conv2D (also known as bottleneck layer).