DeepVision - Group Norm Project

Benedikt Franke

May 30, 2021

1 Theoretical Questions

1.1 What are the key points of the paper?

Wu & He [1] propose a novel normalization layer that divides feature maps into groups and normalizes them group-wise, called Group Normalization (GN). This is motivated by the observation that the often used Batch Normalization (BN) layer gets less powerful the smaller the batch size gets. The reason for this lies in the estimation of mean and variance per batch, which gets less accurate with smaller batch sizes. This poses a challenge for very large models that can only be trained with small batches due to hardware constraints.

GN can be seen as a generalization of Instance Norm (IN) [2] and Layer Norm (LN) [3]. It can simply be inserted into existing models as an replacement for Batch Normalization with only a small loss in performance on bigger batch sizes, and performance gains for smaller batches. It is even possible to fine-tune a pretrained model that used BN for pre-training while switching to GN for the fine-tuning process.

1.2 Which of the lecture topics does the paper relate to?

This paper relates strongly to Part 4 "Training Deep Network Architectures". It describes a normalization technique, and was covered in this chapter. It also addresses the instability problem when using small batch sizes while training deep convolutional networks, which is also covered in this chapter. GN can also be seen as a generalization of IN and LN, which were discussed in this part of the lecture.

As the experiments were done in the context of ResNet on ImageNet, this paper also relates strongly to Part 5 "Deep Network Architectures and Vision Applications", where architectures for image recognition were discussed and Part 3 "DCNN - Concepts and Components", where the concept of a deep convolutional network was introduced, as well as the ImageNet challenge and normalization layers.

Part 5 additionally relates to the paper, as Group Norm was also experimented with in the context of Object Detection, which was also discussed in this chapter.

Naturally, the paper relates to the first two parts of the lecture as the basics of supervised learning, classification and neural networks were covered there.

1.3 Based on the knowledge that you acquired in the lecture, what is the position of the paper in the literature?

As an alternative to BN, it naturally competes with the works on IN and LN. However, according to the experiments in [1], GN beats IN and LN while also being a flexible generalization of those two.

GN's advantage over BN on smaller batch sizes could mean that it will get more important as models get bigger and bigger in settings where hardware constraints forbid the training with large batches. This advantage could, however, be offset by the findings (see [1], Section 5) that GN's performance degrades in heavily parallel settings.

It is possible that GN has applications in Transformer models and recurrent networks, as GN is a generalization of LN which is used in these network types [3].

A major disadvantage for the proposed normalization is that most established architectures use BN, which puts practitioners wanting to use GN for their experiments at the disadvantage of having to adapt existing architectures to use GN. This could of course change if the popularity of GN rises.

1.4 Which issues remain open and should be addressed in future work?

The paper did not provide a theoretical investigation of the optimal choice of the hyperparameter G. Instead, only a set of values were evaluated for the ImageNet-dataset. A more general rule of thumb how G should be chosen would make it easier to adapt GN for other experiments. The need to grid search multiple values of G increases the complexity of adapting GN when compared to BN, as BN does not require the tuning of additional hyperparameters and instead works 'out of the box'.

Furthermore, the paper did not - like stated in Section 6 [1] - study the potential of GN models when the architecture is designed from the ground up with GN in mind. In the current experiments, all models were built with BN and only adapted to use GN. Therefore, the question about the maximum possible performance of GN-based models remains unanswered.

Last but not least, the paper did not investigate potential applications of GN outside of the convolutional domain, e.g. in recurrent networks where LN is used [3].

2 Motivation of Methodology

The basic methodology of [1] was to take a working setup to train ResNet50 on ImageNet, then replace BN with GN and observe the differences. For this, they used the ImageNet-training procedure described in [4].

To keep close to the general procedure, while using another network and another dataset, I decided to follow the CIFAR-10 training procedure and ResNet-architecture described in [5], Section 4.2. I selected the ResNet20-architecture from the described networks with the adaptions for CIFAR-10, which mostly consist of a lower amount of feature maps and no intermediate pooling layers compared to ResNet for ImageNet. My only deviation from the described architecture consists in the residual connections. While in [5], only identity shortcuts where used to keep the number of parameters equal to their non-residual baseline, I used projection shortcuts when dimensions change between layers. The experiments in [5] showed projection shortcuts to be beneficial, and I do not have a baseline I need to match in parameters.

For the parameter G of the GroupNorm layer, [1] used G = 32 in all cases. As the CIFAR10-ResNet has layers with only 16 feature maps, I suspected $G = 8$ to be the correct choice. This would keep the relationship that G is equal to half of the minimum amount of feature maps in the network. A short parameter search over $G \in \{8, 4, 2\}$ confirmed this. I deliberately did not test values of G that would transform some GN layers into IN or LN, i.e. $G = 16$ or $G = 1$.

For the final experiments after the parameter search, I deviated from the training procedure of [5] slightly, as I train for 100 epochs (with variable batch size for the experiments) instead of 64k iterations with a fixed batch size of 128. The number of epochs, and the learning rate schedule were again taken from [1], which is rather similar to [5] (dividing the learning rate by 10 at evenly spaced points of the training).

While [1] evaluated their models on center 224 x 224 crops of ImageNet and the median of the final 5 epochs' validation error were taken, the evaluation for my experiments was done on the plain, unmodified test set of CIFAR-10 like in [5]. As I was sceptical of the effectiveness of taking the median of the final 5 epochs' test score to combat random variations, I instead repeated each experiment 5 times in order to average the final test score over multiple initializations. After all, a model state at time t depends on all previous time steps, which all depend on the initialization of the network. Each run was done using a different seed for the random number generator. However, even when running multiple times the trend described in [1] remains clearly visible.

An visualization of the used architecture can be seen in Figure 2.

3 Results

As visualized in Figure 1, the effect described in [1] could be reproduced here. In contrast to the ImageNet-experiments in [1], Group Normalization "overtakes" Batch Norm one batch size later in my experiments on CIFAR-10. Still, the trend that Batch Normalization loses regularization capability while Group Norm remains stable when batch sizes get smaller remains clearly visible. The error of the networks trained with Group Norm remains approximately constant, while the error of the Batch

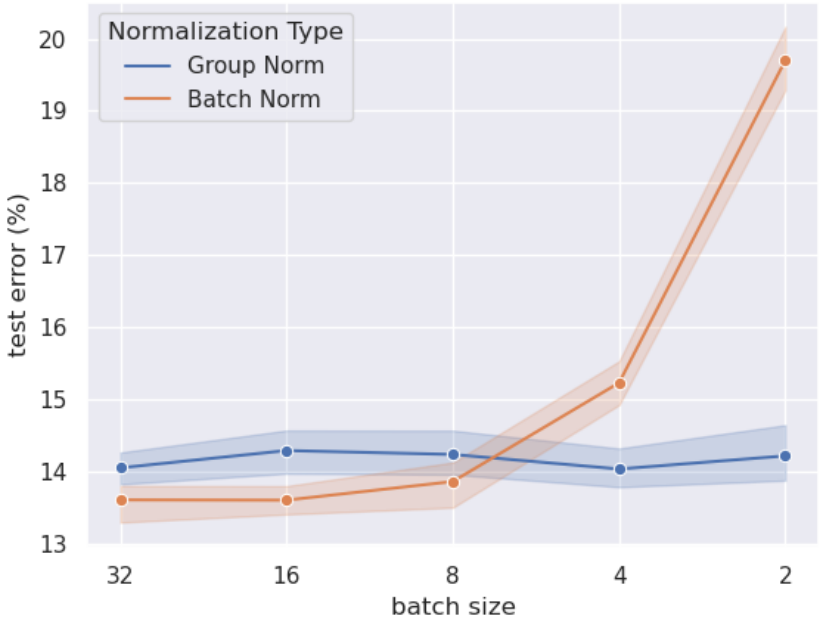

Figure 1: Reproduction of Figure 1 from [1], shaded area is 95% CI for data from runs with 5 different seeds

Norm-based networks rises rapidly for batch sizes smaller than 8. Therefore, my experiments support the claim that Group Normalization is robust against small batch sizes made in [1].

References

[1] Y. Wu and K. He, "Group normalization," 2018.

[2] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," 2017.

[3] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016.

[4] S. Gross and M. Wilber, "Training and investigating residual nets," 2016. https://github.com/facebook/fb.resnet.torch.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

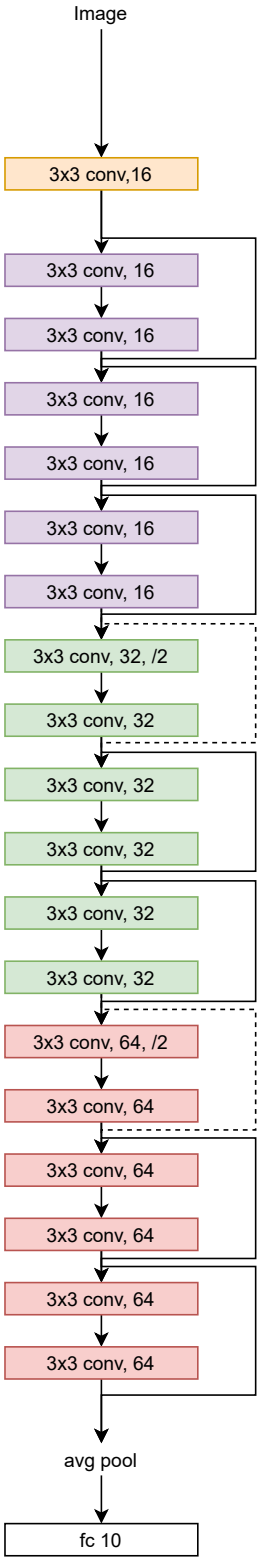

Figure 2: ResNet20-architecture used for my experiments