Deep Learning is not a Matter of Depth but of Good Training

1st Björn Barz
Computer Vision Group
Friedrich Schiller University Jena
Jena, Germany
bjoern.barz@uni-jena.de

2nd Joachim Denzler
Computer Vision Group
Friedrich Schiller University Jena
Jena, Germany
joachim.denzler@uni-jena.de

***Abstract*—In the past few years, deep neural networks have often been claimed to provide greater representational power than shallow networks. In this work, we propose a wide, shallow, and strictly sequential network architecture without any residual connections. When trained with cyclical learning rate schedules, this simple network achieves a classification accuracy on CIFAR-100 competitive to a 10 times deeper residual network, while it can be trained 4 times faster. This provides evidence that neither depth nor residual connections are crucial for deep learning. Instead, residual connections just seem to facilitate training using plain SGD by avoiding bad local minima. We believe that our work can hence point the research community to the actual bottleneck of contemporary deep learning: the optimization algorithms.**

***Index Terms*—deep learning, stochastic gradient descent, learning rate schedule, residual networks, cyclical learning rates**

I. Introduction

In the past few years, deep learning using convolutional neural networks (CNNs) has continuously advanced the state-of-the-art in most image processing and other machine learning applications: The first notable success of modern deep learning was in 2012, when a CNN often called *AlexNet* [1] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). This network, which is composed of 8 trainable hidden layers, was outperformed two years later by the so-called *VGG* architecture [2], which roughly doubles the number of trainable layers to between 16 and 19.

Likewise, VGG was surpassed in 2015 by deep residual networks (*ResNets*) [3] with up to 152 trainable layers. To avoid vanishing gradients during backpropagation through all those layers, ResNet contains skip-connections that add the output of earlier layers to that of some later layers. On smaller benchmark datasets such as CIFAR-100 [4], where training does not take weeks, the trend of increasing the number of hidden layers has continued and lead to ResNets with up to 1001 layers [5].

The paradigm of creating CNNs as deep as possible arises from the fact that the degree of abstraction of the learned image features generally increases with the depth of the network: The first layer learns to recognize simple visual features such as edges, while activations of later layers often refer to semantic object parts [6]. Therefore, it is often postulated that the abstraction and generalization capability of CNNs is a monotonically increasing function of its depth, *i.e.*, the number of trainable layers [2], [3], [5].

However, there also are other important properties of CNNs such as their width (*i.e.*, the number of channels per layer) and the optimization algorithm used during training. Both VGG and ResNet have originally been trained using stochastic gradient descent (SGD) with momentum, starting with a manually selected learning rate that is reduced by a factor of 10 each time the performance on a held-out validation set reaches a plateau [2], [3].

In this work, we show that a shallow but wide network with as few as 11 trainable layers and *no* residual connections can achieve the same performance on CIFAR-100 as a thin deep residual network with 110 parametric layers when trained using a more sophisticated learning rate schedule. Moreover, our network requires less run-time during inference and training due to its reduced depth.

This finding provides evidence that the depth of a CNN is less important than commonly thought. Intuitively, a certain depth is required for sufficient abstraction capability, but more layers do not seem to contribute significantly to this aspect.

The main contributions of our work are as follows:

1) We propose a novel strictly sequential CNN architecture with 11 layers that can achieve a performance competitive to a 10 times deeper ResNet-110 on CIFAR-100.
2) We compare the effects of two recent learning rate schedules (CLR and SGDR) on sequential and residual networks.
3) This leads to the insight, that ResNets are not necessarily much more powerful than wide shallow sequential networks, but just easier to train with plain SGD.

Although our experiments are not meant to beat the current state-of-the-art in image classification, we are convinced that this is an interesting finding and hope that our work can help pointing the research community to the real bottleneck of deep learning, which is neither insufficient depth nor a lack of architectural complexity, but poor optimization algorithms.

This work was supported by the German Research Foundation as part of the priority programme "Volunteered Geographic Information: Interpretation, Visualisation and Social Computing" (SPP 1894, contract DE 735/11-1).

II. Related Work

The effects of depth and width of residual networks on their performance have already been investigated by Zagoruyko and Komodakis [7], who have shown that increasing the width of ResNets while decreasing their depth improves classification performance. They state "that the main power of deep residual networks is in residual blocks, and that the effect of depth is supplementary" [7, p. 3].

Our findings take this insight a step further and suggest that the residual connections do not increase the representational power of the network either, but just facilitate training using plain SGD. However, strictly sequential wide shallow networks without residual connections can achieve similar performance when trained with more sophisticated optimization algorithms.

Our work is furthermore similar in spirit to the analysis of Melis et al. [8], who have shown that rather shallow LSTMs can outperform deeper and more recent neural language models thanks to thorough hyper-parameter tuning and regularization.

We present analogous findings for the computer vision domain, but achieve this by focusing on sophisticated learning rate schedules instead of hyper-parameters.

III. Network Architecture

The convolutional part of our wide shallow CNN is composed of 4 sequential blocks of convolutional units, as outlined in Fig. 1. Each unit consists of a convolutional layer followed by a ReLU activation and batch normalization [9].

The first block consists of 2 units with 64 channels each, the second and third block of 3 units with 128 and 256 channels, respectively, and the last block of a single unit with 512 channels. Sub-sampling by a factor of 2 along the spatial axes is performed between the blocks using average pooling and the last block is followed by global average pooling.

The subsequent fully-connected part of the network consists of a fully-connected layer with 512 channels followed by ReLU and batch normalization, and a final fully-connected layer with softmax activation, whose number of channels equals the number of classes.

This architecture is similar to the VGG networks [2], but with a few important differences:

1) We use batch normalization [9] between layers to mitigate the problem of exploding and vanishing gradients.
2) We use average pooling instead of maximum pooling between convolutional blocks.
3) Like ResNet [3], we use global average pooling after the last convolutional layer instead of fully-connected pooling. This enables the network to handle input images of varying size.

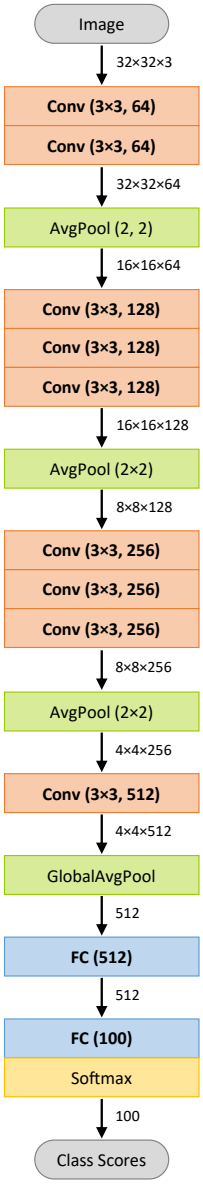

Fig. 1. Architecture of our wide shallow sequential network. All convolutional and the first fully-connected layer are followed by a ReLU activation and Batch Normalization, which we omitted for clarity. Trainable layers are set in bold font.

IV. Optimization

Both VGG [2] and ResNet [3] have originally been trained using SGD with momentum and a simple learning rate schedule: Starting with a small initial learning rate to avoid exploding gradients ("warm-up phase"), the learning rate is set to a higher value after a few iterations and then decreased by a factor of 10 either after a handpicked number of iterations or when the performance on a held-out validation set has not improved significantly during the last few epochs.

However, this approach is prone to getting trapped in bad local minima, since the learning rate is only decreased, but never increased. Two approaches have recently been proposed for overcoming this issue by periodically increasing and decreasing the learning rate. These are reviewed in the following subsections and depicted in Fig. 2, along with the schedule that He et al. [3] used for training ResNet-110.

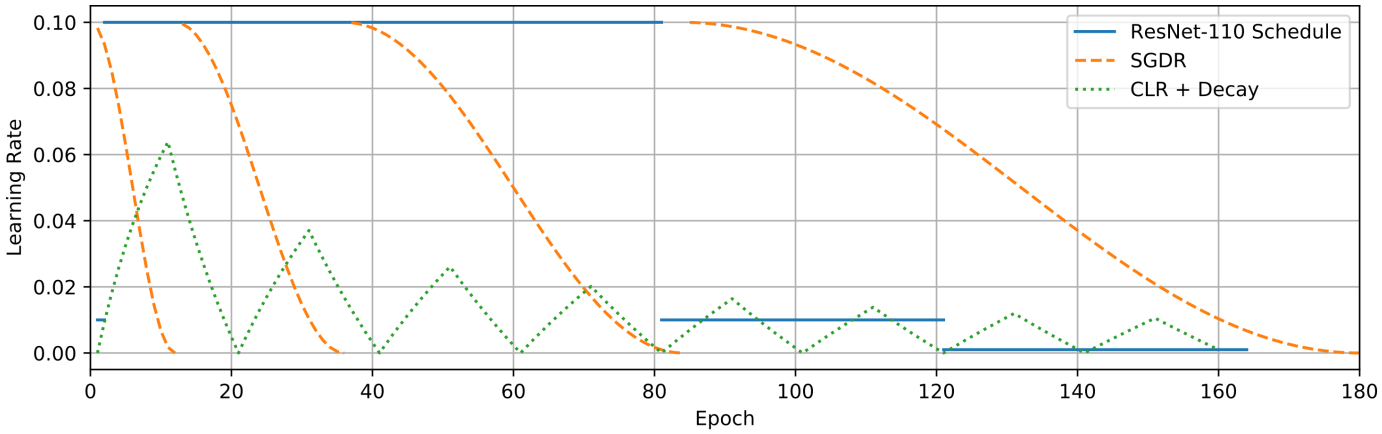

Fig. 2. Different Learning Rate Schedules

A. Cyclical Learning Rates (CLR)

Cyclical Learning Rates (CLR) [10] start with a small base learning rate $\mathrm{lr}_{\min}$ and increase it linearly during a fixed number s of iterations ("step size") up to a certain maximum learning rate $\mathrm{lr}_{\max}$. Thereafter, the learning rate is decreased again over the same number of iterations until reaching the base learning rate. After that, the next cycle of increasing and decreasing the learning rate begins. Formally, the learning rate $\mathrm{clr}(t)$ at iteration t is

$$\mathrm{clr}(t) = \mathrm{lr}_{\min} + (\mathrm{lr}_{\max} - \mathrm{lr}_{\min}) \cdot \left(1 - \left|\frac{t}{s} - 2\left\lfloor\frac{t}{2s}\right\rfloor - 1\right|\right) . \quad (1)$$

The motivation behind increasing the learning rate from time to time is to enable the learning process to escape from bad local minima.

The authors also proposed a variant where the maximum learning rate is reduced after each cycle. In this work, we achieve this in a slightly different way by applying a global learning rate decay so that the maximum learning rate at the end of the training is δ times lower than the initial maximum learning rate:

$$\widetilde{\mathrm{clr}}(t) = \frac{\mathrm{clr}(t)}{1 + (\delta - 1) \cdot \frac{t}{t_{\max}}} , \quad (2)$$

where $t_{\max}$ is the maximum number of iterations after which the training procedure is terminated.

B. Stochastic Gradient Descent with Warm Restarts (SGDR)

A similar approach is taken by SGD with Warm Restarts (SGDR) [11]. In contrast to CLR, the learning rate is not decreased linearly but according to cosine annealing. It is furthermore not increased smoothly at the end of each cycle, but in an instant. Formally, the learning rate $\mathrm{sgdr}(\epsilon)$ during epoch ϵ is

$$\mathrm{sgdr}(\epsilon) = \mathrm{lr}_{\min} + \frac{1}{2}(\mathrm{lr}_{\max} - \mathrm{lr}_{\min})\left(1 + \cos\left(\frac{\epsilon_i}{S_i}\pi\right)\right) , \quad (3)$$

where S_i is the length of the current cycle and ϵ_i is the number of epochs passed since the beginning of the cycle.

Starting with a certain initial cycle length S_0, the length S_i of cycle i results from multiplication of the length of the previous cycle with a constant factor:

$$S_i = \sigma \cdot S_{i-1} . \quad (4)$$

Since SGDR starts with a very high learning rate, we combine it with gradient clipping [12] to avoid exploding gradients: The maximum norm of the gradients propagated back through the network is restricted to 10.0.

V. Experiments

A. Setup

We compare the performance of our strictly sequential VGG-like network architecture and deep residual networks (ResNets) trained with different learning rate schedules on the CIFAR-100 dataset. CIFAR-100 is a heavily benchmarked dataset consisting of images from 100 different classes. Each class comprises 500 training and 100 test images, leading to a total amount of 50k training and 10k test images. All images are in RGB color format and of size 32×32 pixels.

We compare 3 different learning rate schedules for training our plain 11-layer network, referred to as "Plain-11" in the following, and ResNet-110, which has ten times more parametric layers:

- The handcrafted learning rate schedule used by He et al. [3] for training ResNet-110: Starting with a learning rate of 0.01 to avoid divergence, the learning rate is increased to 0.1 after the first epoch and then divided by 10 after 80 and 120 epochs. Training is terminated after a total of 164 epochs.
- CLR with a step size s corresponding to 10 epochs (*i.e.*, s is the number of batches per epoch multiplied with 10), a minimum learning rate of $\mathrm{lr}_{\min} = 10^{-5}$, a maximum learning rate of $\mathrm{lr}_{\max} = 0.1$ and a final learning rate decay of $\delta = 10$.

TABLE I
CIFAR-100 VALIDATION ERROR OF DIFFERENTLY TRAINED CNNS.

Architecture	Time / Epoch	Schedule	Epochs	Err. Rate
ResNet-110	175 s	SGD	164	26.51 %
		CLR	160	27.78 %
		SGDR	180	25.98 %
Plain-11 (ours)	42 s	SGD	164	28.48 %
		CLR	160	27.26 %
		SGDR	180	27.09 %

- SGDR with a base period length of $S_0 = 12$ epochs, which is doubled after each period ($\sigma = 2$). The minimum learning rate is $\mathrm{lr}_{\min} = 10^{-6}$ and the maximum learning rate is $\mathrm{lr}_{\max} = 0.1$.

The number of training epochs for CLR and SGDR has been chosen to be near 164 for being comparable with the ResNet schedule, but so that the learning rate arrives at $\mathrm{lr}_{\min}$ during the last epoch. This is important, because the performance of the network often behaves unstable while training with higher learning rates.

We use a mini-batch size of 100 images—as opposed to the most commonly used batch-size of 128—because it divides the total amount of training images evenly.

B. Comparison of Architectures

For all combinations of network architectures and learning rate schedules, we trained 3 networks with different random weight initializations and report the average error rate on the test set in Table I. It can be seen that our strictly sequential network does not achieve the performance of ResNet-110 when trained with the same hand-tuned learning rate schedule. Using SGDR, however, improves its validation error rate by 5%, while the improvement for ResNet-110 is only 2%. This indicates that the residual connections used in ResNet already facilitate training, so that different learning rate schedules do not add much benefit.

On the other hand, more sophisticated training procedures allow us to train a shallow sequential network that achieves competitive performance compared to ResNet-110. At the same time, the architecture of our model is less complex and can be trained 4 times faster, which is an important aspect in the compute-intensive realm of deep learning.

C. Comparison of CLR and SGDR

It is also worth noting that SGDR achieved a better performance than CLR for both network architectures, while CLR even performed worse than the handcrafted learning rate schedule for training ResNet-110. Of course, our experiments do not include a sufficient number of network architectures and datasets to claim that one method for scheduling learning rates would perform better than the other, but it at least provides some evidence that SGDR is a good choice.

On the other hand, CLR provides more flexibility for choosing the total number of training epochs, since all cycles have the same, comparatively small length. Since the length of the SGDR periods grows exponentially, one cannot just add a few but only a lot more training epochs when using SGDR.

VI. CONCLUSIONS

Wide residual networks [7] have already demonstrated that the depth of a CNN is much less important than previously thought and that a wide shallow ResNet can outperform thin deep ResNets significantly. We have shown that a plain VGG-like network without any residual connections can achieve the same performance as a 10 times deeper ResNet when using sophisticated learning rate schedules such as CLR or SGDR, while being 4 times faster to train.

This provides evidence that the skip-connections used in ResNets do not increase the representational power of the network, but rather lead to loss functions with fewer bad local minima and hence facilitate learning using SGD.

The main problem of modern deep learning is, thus, neither insufficient depth nor a lack of structural complexity, but the deficiencies of the ubiquitous stochastic gradient descent (SGD) optimization algorithm and its many variants, which easily get stuck in bad local minima.

While there is a large amount of ongoing work about new network architectures, types of layers, and activation functions, we believe that a large fraction of the potential of most neural networks is wasted due to poor optimization algorithms. More complex learning rate schedules such as the cyclical learning rates [10], [11] discussed in this paper or the parameter-specific learning rates used by AdaGrad [13], RMSProp [14], and Adam [15] are a step towards getting to the root of the problem, but are still based on SGD and backpropagation.

While the shape of the categorical cross-entropy loss function might be one part of the problem, Nguyen et al. [16] have shown that even in the face of convex loss functions, SGD makes steep improvements in the beginning, but then stagnates at a sub-optimal state. Thus, the next big breakthrough in deep learning might rather be achieved by thinking out of the box and applying completely novel optimization algorithms for training artificial neural networks.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems (NIPS)*, 2012, pp. 1097–1105.

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

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[4] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 630–645.

[6] M. Simon, E. Rodner, and J. Denzler, "Part detector discovery in deep convolutional neural networks," in *Asian Conference on Computer Vision (ACCV)*, 2014, pp. 162–177.

[7] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *British Machine Vision Conference (BMVC)*, 2016.

[8] G. Melis, C. Dyer, and P. Blunsom, "On the state of the art of evaluation in neural language models," *arXiv preprint arXiv:1707.05589*, 2017.

[9] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.

[10] L. N. Smith, "Cyclical learning rates for training neural networks," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2017, pp. 464–472.

[11] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," in *International Conference on Learning Representations (ICLR)*, 2017.

[12] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International Conference on Machine Learning (ICML)*, 2013, pp. 1310–1318.

[13] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[14] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[15] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[16] L. M. Nguyen, N. H. Nguyen, D. T. Phan, J. R. Kalagnanam, and K. Scheinberg, "When does stochastic gradient algorithm work well?" *arXiv preprint arXiv:1801.06159*, 2018.