ReZero is All You Need: Fast Convergence at Large Depth (Supplementary material)

Thomas Bachlechner[*1] **Bodhisattwa Prasad Majumder***[2] **Henry Mao***[3] **Gary Cottrell**[2] **Julian McAuley**[2]

[1]MeetElise, USA, thomas@meetelise.com
[2]Computer Science and Engineering, UC San Diego, USA, {bmajumde, gary, jmcauley}@eng.ucsd.edu
[3]Altum Inc., USA, henry@calclavia.com

A VANISHING SINGULAR VALUES IN TRANSFORMERS

Two crucial components relevant to the signal propagation in the Transformer include LayerNorm [Ba et al., 2016] and (multi-head) self attention [Vaswani et al., 2017]. In this section, we argue that neither component by itself or in conjunction with a vanilla residual connection can satisfy dynamical isometry for all input signals.

A.1 LAYERNORM

Layer normalization removes the mean and scales the variance over all neurons of a given layer and introduces learnable parameters γ and β to re-scale the variance and shift the mean according to

$$\text{LayerNorm}(\boldsymbol{x}) = \frac{\boldsymbol{x} - \text{E}(\boldsymbol{x})}{\sqrt{\text{Var}(x)}} \times \gamma + \beta\,. \tag{1}$$

It is clear from this definition that perturbing an input x by a transformation that purely shifts either its mean or variance will leave the output unchanged. These perturbations, therefore, give rise to two vanishing singular values of the input-output Jacobian. In the Transformer architecture [Vaswani et al., 2017], the norm is applied to each of the n elements of the input sentence, leading to a total of $2 \times n$ vanishing singular values of the Jacobian for each Transformer layer.

A.2 SELF-ATTENTION

Self-attention allows the model to relate content located across different positions by computing a weighted sum of an input sequence. Specifically, the $n \times d$ matrix $\boldsymbol{x}$ contains an input sequence of n rows containing d-dimensional embedding vectors, from which we can evaluate the query, key and value matrices $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} = \boldsymbol{x} \cdot \boldsymbol{W}^{Q,K,V}$, where the $W^{Q,K,V}$ matrices are $d \times d$. The scaled dot-product attention then is given by

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\boldsymbol{Q} \cdot \boldsymbol{K}^\top / \sqrt{d}\right) \cdot \boldsymbol{V}\,. \tag{2}$$

In general, the singular value spectrum of the Jacobian of this attention process is complicated. Rather than studying it in full generality, we now merely argue that for some inputs $\boldsymbol{x}$ and weights $\boldsymbol{W}^{Q,K,V}$ the Jacobian has a large number of vanishing singular values (a claim we evaluate empirically in the paper). Consider weights or inputs such that each of the arguments of the softmax function is small compared to 1. The softmax function then simply returns a $n \times n$ dimensional matrix filled with entries that all approximate $1/n$. This means that the attention function projects all embedding vectors of the input sequence onto a single diagonal direction. This implies that out of the $n \times d$ Jacobian singular values only d are non-vanishing and hence much of the input signal is lost. A residual connection can restore some of the lost signals, but even then some perturbations are amplified while others are attenuated. This example demonstrates that self-attention is incompatible with dynamical isometry and unimpeded signal propagation in deep Transformer networks. It is easy to verify that the same conclusion holds for multi-head attention. A careful initialization of the weights might alleviate some of these issues, but we are not aware of any initialization scheme that would render a Transformer layer consistent with dynamical isometry.

B CONVERGENCE SPEED EXPERIMENTAL HYPERPARAMETERS

For all model variants in Section 6.2, we control the batch size to be 1080, number of layers to 12, feed-forward and attention dropout to 20%, hidden and embedding size to 512 units, context length to 512, the attention heads to 2, and GELU [Hendrycks and Gimpel, 2016] activation in the

*Authors contributed equally, ordered by last name

Supplement for the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2021).

Model	Iterations	Speedup
Post-Norm	Diverged	-
+ Warm-up	13,690	$1\times$
Pre-Norm	17,765	$0.77\times$
GPT2-Norm	21,187	$0.65\times$
ReZero $\alpha = 1$	14,506	$0.94\times$
ReZero $\alpha = 0$	**8,800**	**1.56**$\times$

Table 1: Comparison of various 12 layer Transformers normalization variants (Post-Norm [Vaswani et al., 2017], Pre-Norm, GPT2-Norm [Radford et al., 2019]) against ReZero and the training iterations required to reach 1.2 BPB on `enwiki8` validation set.

point-wise feed-forward layer. To accommodate large batch training we use the LAMB optimizer [You et al., 2019] with a fixed learning rate of 0.016. Although learning rate schedules tend to improve performance [Devlin et al., 2019], we omit them to simplify our training process. Training is performed on 8x V100 GPUs for at most a few days. Table 1 shows the speed gain of ReZero to reach 1.2 BPB as compared to Transformers normalization variants. The findings is also reflected in Figure 7.

C DEEP TRANSFORMERS EXPERIMENTAL HYPERPARAMETERS

In Section 6.3, in order to examine whether our approach scales to deeper Transformers, we scale our 12 layer ReZero Transformer from Section 6.2 to 64 layers and 128 layers and compare it against the vanilla Transformer (*Post-Norm*). Due to memory constraints, we decreased the hidden size from 512 to 256 and reduced batch size to 304 and 144 for the 64 layer and 128 layer model respectively. Following guidelines from [You et al., 2019] we also adjusted the learning rate according to $0.0005 \times \sqrt{\text{batch size}}$. For all models in our experiments we limit training to a maximum of 100 training epochs. Training is performed on 8x V100 GPUs for at most a few days. Table 1

D REVIEW OF RESIDUAL GATES FOR DEEP SIGNAL PROPAGATION

In this section we give a chronological but light review of residual gates that are designed to preserve signals as they propagate deep into neural networks.

D.1 HIGHWAY NETWORKS

Highway Networks Srivastava et al. [2015], based on ideas from LSTM Hochreiter and Schmidhuber [1997], were the first feedforward neural networks with hundreds of layers. This architecture employs gating units that learn to regulate signal flow through the network. Specifically, the authors define transform and carry gates $T[\boldsymbol{W}_T](\boldsymbol{x})$ and $C[\boldsymbol{W}_C](\boldsymbol{x})$, with weights $\boldsymbol{W}_{T,C}$ that act explicitly non-linearly on the signal $\boldsymbol{x}$. When combined with some block $F(\boldsymbol{x}_i)$ of a deep network, this gives the transformation

$$\boldsymbol{x}_{i+1} = C[\boldsymbol{W}_C](\boldsymbol{x}) \cdot \boldsymbol{x}_i + T[\boldsymbol{W}_T](\boldsymbol{x}) \cdot F(\boldsymbol{x}_i)\,. \tag{3}$$

The authors further propose to simplify the architecture according to $C \equiv 1 - T$, and using a simple Transform gate of the form $T[\boldsymbol{W}_T](\boldsymbol{x}) \equiv \sigma(\boldsymbol{W}_T^\top \cdot \boldsymbol{x} + \boldsymbol{b}_T)$, where σ denotes some activation function. The bias is initialized to be negative, as to bias the network towards carry behavior, e.g., C, but the network is not initialized as the identity map.

The proposal of Highway Networks lead to Gated ResNet Savarese et al. [2016], in which there exists a single additional parameter that parametrizes the gates as $\boldsymbol{W}_T = \boldsymbol{0}$, $\boldsymbol{b}_T = \alpha$, $C = 1 - T$.

Any feed-forward network could be written in the form (13), and ReZero corresponds to the simple choice $\boldsymbol{W}_T = \boldsymbol{W}_C = \boldsymbol{0}$, $\boldsymbol{b}_T = \alpha$, $\boldsymbol{b}_C = 1$. In contrast to Highway Networks, in ReZero the gates are independent of the input signal. We compare the performance of Gated ResNets to ReZero ResNets in Section 5.

D.2 RESNETS

ResNets He et al. [2016a] introduced the simple residual connection

$$\boldsymbol{x}_{i+1} = \sigma\left(\boldsymbol{x}_i + F(\boldsymbol{x}_i)\right)\,, \tag{4}$$

that has been extremely successful in training deep networks. Just as Highway Networks, these residual connections are not initialized to give the identity map.

D.3 PRE-ACTIVATION RESNETS

Soon after the introduction of ResNets it was realized in He et al. [2016b] that applying the activation function σ prior to the residual connection allows for better performance. Schematically, we have the pre-activation connection

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + F(\boldsymbol{x}_i)\,, \tag{5}$$

where we absorbed the activation function into the block $F(\boldsymbol{x}_i)$. This finding of improved performance is consistent with improved signal propagation, since the residual connection is not modulated by the activation function.

D.4 ZERO γ

Residual networks often contain normalization layers in which the signal is rescaled by learnable parameters γ which

Model	Val. Error [%]	Change	Epochs to 80% Acc.	Train Loss $\times 1000$
ResNet-56	6.27 ± 0.06	–	20 ± 1	5.9 ± 0.1
+ Gated ResNet	6.80 ± 0.09	$+ 0.53$	9 ± 2	4.6 ± 0.3
+ zero γ	7.84 ± 0.05	$+ 1.57$	39 ± 4	31.2 ± 0.5
+ FixUp	7.26 ± 0.10	$+ 0.99$	13 ± 1	4.6 ± 0.2
+ **ReZero**	6.58 ± 0.07	$+ 0.31$	15 ± 2	4.5 ± 0.3
ResNet-110	6.24 ± 0.29	–	23 ± 4	4.0 ± 0.1
+ Gated ResNet	6.71 ± 0.05	$+ 0.47$	10 ± 2	2.8 ± 0.2
+ zero γ	7.49 ± 0.07	$+ 1.25$	36 ± 5	18.5 ± 0.9
+ FixUp	7.10 ± 0.22	$+ 0.86$	15 ± 1	3.3 ± 0.5
+ **ReZero**	5.93 ± 0.12	$- 0.31$	14 ± 1	2.6 ± 0.1
Pre-activation ResNet-18	6.38 ± 0.01	–	26 ± 2	4.1 ± 0.3
+ **ReZero**	5.43 ± 0.06	$- 0.95$	12 ± 1	1.9 ± 0.3
Pre-activation ResNet-50	5.37 ± 0.02	–	26 ± 3	2.6 ± 0.1
+ **ReZero**	4.80 ± 0.08	$- 0.57$	17 ± 1	2.2 ± 0.1

Table 2: Comparison of ResNet variants on CIFAR-10. The uncertainties correspond to standard error. Baselines used: Gated ResNet [Srivastava et al., 2015, Savarese et al., 2016], zero γ [Goyal et al., 2017, Hardt and Ma, 2016], FixUp [Zhang et al., 2019], ResNet-56/110 [He et al., 2016a], Pre-activation ResNet-18/50 [He et al., 2016b]

is referred to the Zero γ Goyal et al. [2017], Hardt and Ma [2016], He et al. [2019]. If the last element before a residual connection happens to be a normalization layer, then initializing these γ to zero has been found to improve convergence speed and accuracy. This method is in spirit very similar to the ReZero architecture. However, it potentially zero-initializes many parameters for each block, and is only applicable when a normalization layer exists.

D.5 FIXUP

FixUp initialization Zhang et al. [2019] carefully rescales the initialization scheme in order to avoid vanishing or exploding gradients, without the use of normalization techniques. In particular, this scheme is implemented via the following Rules (verbatim from Zhang et al. [2019]):

1. Initialize the classification layer and the last layer of each residual branch to 0.

2. Initialize every other layer using a standard method (e.g., [He et al., 2015]), and scale only the weight layers inside residual branches by $L^{-1/(2m-2)}$.

3. Add a scalar multiplier (initialized at 1) in every branch and a scalar bias (initialized at 0) before each convolution, linear, and element-wise activation layer.

The authors emphasize that Rule 2 is the essential part. ReZero is simpler and similar to the first part of Rule 3, but the initialization differs.

D.6 SKIPINIT

De and Smith [2020] proposes to replace BatchNorm layers with a single scalar initialized at a small value. SkipInit is only applicable when normalization layers exist.

D.7 REZERO

ReZero is the simplest iteration achieving the goal of deep signal propagation. Schematically, the ReZero architecture is

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i F(\boldsymbol{x}_i)\,. \tag{6}$$

The rule to implement ReZero is

- For every block add a scalar multiplier α (initialized at 0) and a residual connection.

E CIFAR-10 EXPERIMENTS

In this section we briefly describe the hyperparameters used in the image recognition experiments performed in §5. For all these experiments we used PyTorch version 1.2.0 (we have observed some inconsistencies in results with other PyTorch versions that may be due to different default initializations). CIFAR10 experiments tend to take less than an hour on a single RTX 2080TI GPU. The comparison for various ResNet variants is in Table 2.

E.1 STEP-DOWN SCHEDULE

In Table 1 we compare the inference accuracy of different network architectures after training with identical hyper-

parameters a learning-rate schedule that decreases in three steps, as in He et al. [2016a]. The initial learning rate is 0.1 and decreases by a factor of 10 at 100 and 150 epochs. The models are trained for a total of 200 epochs. We use the SGD optimizer with a batch size of 128, weight decay of 5×10^{-4} and momentum 0.9.

E.2 SUPERCONVERGENCE SCHEDULE

To demonstrate superconvergence we use a one-cycle learning rate schedule, as described in Smith and Topin [2019] and closely follow the setup by Fast AI referenced in the text. In particular, the learning rate of the SGD optimizer evolves as follows over 45 epochs. The initial learning rate is 0.032 and linearly increases with each iteration to reach 1.2 after 10% of the total number of iterations has been reached. Then, the learning rate linearly decreases to return to 0.032 when 90% of the total steps. Thereafter, the learning rate linearly decays to a final value of 0.001 at the end of training. The SGD momentum varies between 0.85 and 0.95, mirroring the triangular learning rate, as is standard for the one-cycle policy in this setup Smith and Topin [2019]. Weight decay is set to 2×10^{-4} and the batch size is 512. It was important to have a small, constant learning rate for the residual weights, otherwise the ReZero model diverges easily.

The residual weights cannot tolerate the extremely large learning rates required for the super-convergence phenomenon. For this reason we keep the learning rate of the residual weights at 0.1 throughout training.

F DATASETS

We note that for all our experiments, we follow the official training, validation and test splits of the respective datasets.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Soham De and Samuel L Smith. Batch normalization biases deep residual networks towards shallow paths. *arXiv preprint arXiv:2002.10444*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *NAACL*, 2019. doi: 10.18653/v1/n19-1423. URL `https://doi.org/10.18653/v1/n19-1423`.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Moritz Hardt and Tengyu Ma. Identity matters in deep learning. *arXiv preprint arXiv:1611.04231*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*. Springer, 2016b.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, 2019.

Dan Hendrycks and Kevin Gimpel. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016. URL `http://arxiv.org/abs/1606.08415`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8), 1997.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

Pedro HP Savarese, Leonardo O Mazza, and Daniel R Figueiredo. Learning identity mappings with residual gates. *arXiv preprint arXiv:1611.01260*, 2016.

Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, page 1100612. International Society for Optics and Photonics, 2019.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. Reducing BERT pre-training time from 3 days to 76 minutes. *CoRR*, abs/1904.00962, 2019. URL `http://arxiv.org/abs/1904.00962`.

Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019.