NormFormer: Improved Transformer Pre-training with Extra Normalization

Abstract

During pre-training, the Pre-LayerNorm transformer suffers from a gradient magnitude mismatch: gradients at early layers are much larger than at later layers. The mismatch can be alleviated by the addition of normalization operations at 3 positions inside each layer. This extra normalization incurs negligible compute cost (+0.4% parameter increase), but improves pre-training perplexity and downstream task performance for both causal and masked language models of multiple sizes. Adding NormFormer on top of the GPT3-Medium architecture can reach the SOTA perplexity 40% faster, or converge 0.45 perplexity better in the same amount of time. For masked language modeling, NormFormer improves fine-tuned GLUE performance by 1.9% on average.

1 Introduction

There has been considerable recent interest in improving the speed and performance of transformer (Vaswani et al., 2017) model pre-training. One such innovation is Layer Normalization ("LN"). LN was originally positioned after the residual connection in each layer to reduce the variance in the magnitude of the inputs to each attention layer. One effect of this ("Post-LN") positioning is much larger magnitude gradients at later layers than at earlier layers. Moving the LN operation before the residual connection, ("Pre-LN"), allows larger learning rates and shorter learning rate warmup without sacrificing performance (Xiong et al., 2020). For this reason, most recent large pre-trained language models use Pre-LN (Raffel et al., 2020; Brown et al., 2020; Lieber et al., 2021; Baevski & Auli, 2019; Radford et al., 2019). In this work we show that, while Pre-LN improves stability over Post-LN, it has the opposite side effect: gradients at earlier layers are now much larger than at later layers. We propose `NormFormer`, which alleviates the gradient magnitude mismatch by adding 3 extra normalization operations to each layer. These operations reduce gradients to early layers and increase gradients to later layers, bringing their magnitudes closer together. Compared to well-tuned Pre-LN baselines, `NormFormer` models reach target pre-training perplexities faster and converge to better pre-training perplexities and downstream task performance.

Section 2 describes the proposed modifications, Section 3 shows pre-training and downstream task performance for fully trained `NormFormer` models against well-tuned baselines. Section 4 shows the gradient mismatch introduced by Pre-LN and how `NormFormer` partially alleviates it. Section 5 shows that `NormFormer` improves over the baseline at a wide range of hyperparameter configurations and model sizes, and that removing any of the 3 added operations degrades performance.

2 Approach

LayerNorm In the initial transformer "Post-LN" formulation , one Layer Normalization ("LN") was placed outside each residual block, to improve training efficiency (Vaswani et al., 2017; Ba et al., 2016). LN normalizes its inputs and scales them element wise:

$$LayerNorm(x) = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} \cdot \gamma + \beta \tag{1}$$

where γ and β are trainable parameters, and ϵ is a small constant.

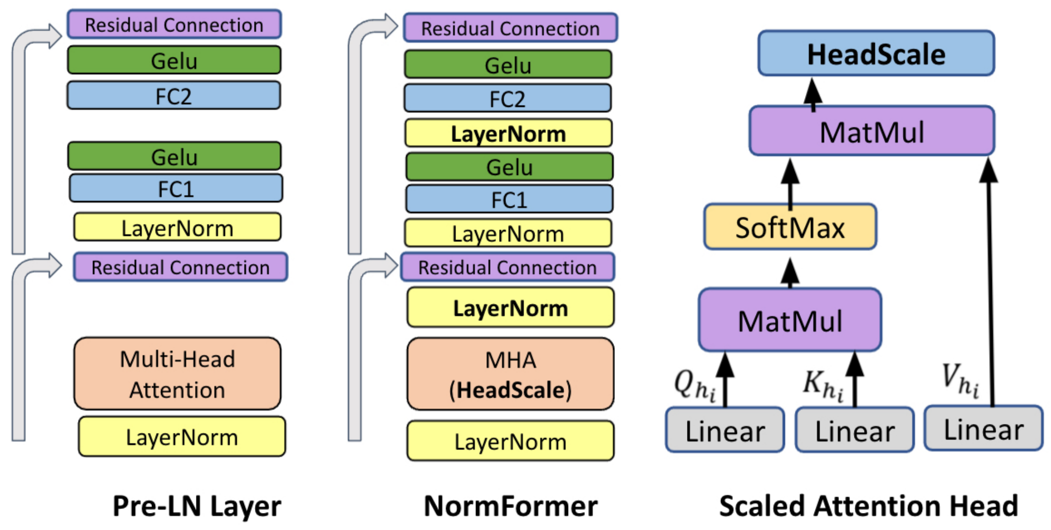

Figure 1: Description of proposed architecture changes. The left block shows the baseline, a Pre-LayerNorm transformer layer. The center block shows NormFormer, with the three additions we propose in bold. The right block zooms in on on a single attention head, with the proposed `HeadScale`.

2.1 NormFormer

We add three operations to each transformer layer: (1) head-wise scaling inside the attention module, (2) LN after the attention module, and (3) LN after the first fully connected layer. The changes are visualized in Figure 1 and described mathematically below.

Scaling Attention Heads The `HeadScale` operation scales the outputs of each attention head linearly.

A multi-head attention module (Vaswani et al., 2017) is originally formulated as:

$$\begin{aligned}&\text{MultiHeadAttention}(Q,K,V) = \text{Concat}(\text{h}_1,\dots,\text{h}_n)W^O\\&\text{h}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V),\end{aligned} \tag{2}$$

where n is the number of heads, i is the index of an attention head, and the projection matrices W^O, W_i^Q, W_i^K, W_i^V are used for output, query, key and value.

At each layer, we introduce a set of n scalar coefficients γ_i.

$$\text{h}_i = \gamma_i \cdot \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \tag{3}$$

2 LNs The two added LNs and follow the attention module. The first LN is inserted after the Scaled Multihead Attention Module, and the second between the two fully connected layers.

In the Pre-LN transformer, our baseline, each layer can be described as

$$Layer(x) = FC_2(FC_1(MHA(LN(x)))) \tag{4}$$

where the FC operations are fully connected (linear) layers with GELU activation (Hendrycks & Gimpel, 2016) and LN is Layer Norm.

Each layer of `NormFormer`, our proposed modification, can be described as

$$NormFormerLayer(x) = FC_2(\mathbf{LN}(FC_1(\mathbf{LN}((\mathbf{Scaled}MHA(LN(x)))) \tag{5}$$

where bolded operations are newly introduced.

All three additions introduce new γ parameters, which provide a cost-effective way to offer each layer an opportunity to change the magnitude of its features, and therefore the magnitude of the gradients to subsequent components.

2.2 EXPERIMENTS

Causal Language Models We pre-train causal LMs ("CLM") that roughly match GPT-3 125M and GPT3-355M proposed in Brown et al. (2020). We show that the conclusions hold for larger models in Section 5.

We differ from Brown et al. (2020) in three ways: (1) we use only dense attention, while they alternate between dense and locally banded sparse attention; (2) we train our models with sinusoidal positional embeddings, following Shortformer Press et al. (2020b), since early experiments found this to produce comparable results with fewer learned parameters; (3) we train on sequences of length 1,024 instead of 2,048 to reduce computational cost, and show that this choice does not change our conclusions in Section 5.

For both the baseline and `NormFormer`, we test six different learning rates `6e-5, 3e-4, 6e-4, 1e-3, 3e-3, 6e-3` for 100,000 updates and pick the best by validation perplexity. Otherwise, we use the same hyperparameters as GPT3-Small and GPT3-Medium, with the linear decay learning rate scheduler. We train the baseline for 300 billion tokens and NormFormer for 282 billion tokens.

Zero Shot Evaluation In addition to validation perplexity, we evaluate CLMs on a subset of the tasks that GPT3 evaluated on in a zero-shot setting (Brown et al., 2020), with the same prompts. We select WinoGrande (Sakaguchi et al., 2020), StoryCloze (**?**), OpenBookQA (**?**), HellaSwag (**?**) and PIQA (Bisk et al., 2020) because GPT3 showed strong performance on these tasks at small scale, as well as consistently improving performance with scale.

Masked Language Models We adopt the Roberta Base, Pre-LN architecture proposed in Liu et al. (2019) without modifying any hyper-parameters. For the baseline, we pre-train for 2 million batches of 1 million tokens, about $\frac{1}{4}$ of the training budget of the original `roberta-base`. NormFormer runs through 192 batches in the same amount of time.

Fine-Tuning We fine-tune both the baseline MLM and NormFormer with learning rates `[1e-5, 1e-4, 3e-4, 1e-3, 3e-3, 6e-3`, and pick the best performing on the validation set for each GLUE task (Wang et al., 2019). Other fine-tuning hyper-parameters follow `roberta-base`.

Pretraining data We pre-train all models on a combination of BOOKCORPUS (Zhu et al., 2019), English Wikipedia, and the English section of CC100 (Conneau et al., 2020) which totals around 300GB of uncompressed text. Pretraining perplexity is computed on the validation subset of this dataset.

Complexity Although the parameter increase is less than 0.07% compared to the baseline, NormFormer introduces 2-6% extra memory and speed cost, with more of a difference for smaller models. 80% of this increase results from the FFN LN, which also accounts for the bulk of the parameter increase. For this reason, subsequent tables compare NormFormer to baseline models trained for the same amount of time on the same hardware, rather than the same number of steps.

3 RESULTS

Figure 2 shows pre-training perplexity curves for CLMs and MLMs, respectively. The green stars, which mark the first validation step where NormFormer crosses the baseline's eventual perplexity, show that NormFormer reaches this target with only 60% and 57% the cost of the baseline, for CLM and MLM respectively.

Figure 2: Pre-Training Perplexity Curves for Causal and Masked Language Models. The green star shows the first validation step where NormFormer crosses the baseline's eventual perplexity, which takes 60% and 57% the wall time for CLM and MLM respectively.

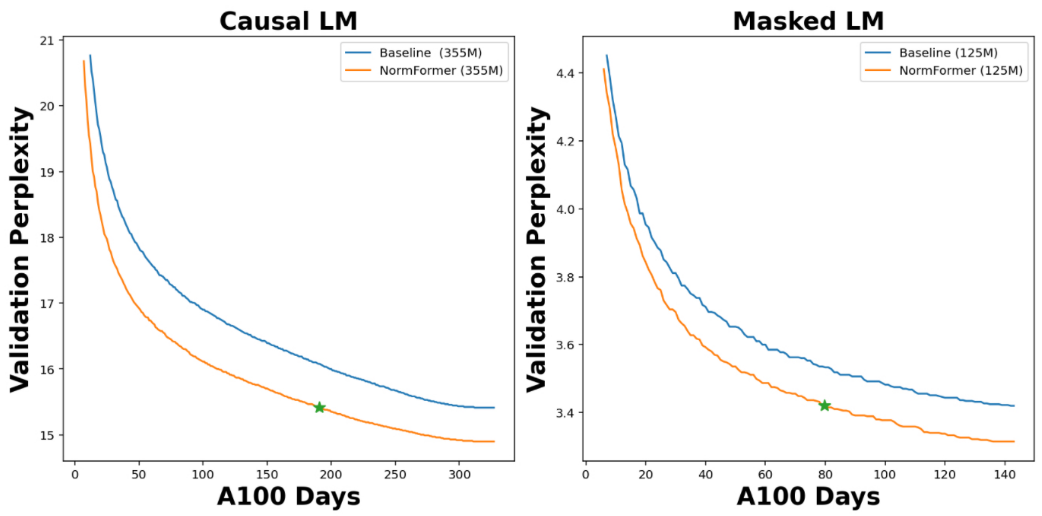

Table 1: Zero-Shot Accuracy for Causal LMs. GPT-3 (paper) results taken from Brown et al. (2020). `HS`: HellaSwag, `PI`: PIQA, `WG`: WinoGrande, `SC`: StoryCloze, `OB`: OpenBookQA. Replicated models are pretrained and evaluated using `fairseq` (Ott et al., 2019). PPL is validation perplexity during pre-training.

	Model Size	PPL	HS	PI	WG	SC	OB	Avg
Random Baseline	-	-	25.0	50.0	50.0	50.0	25.0	40.0
GPT3 (paper)	125.00	-	33.7	64.6	52.0	63.3	35.6	49.8
GPT3 (replicated)	124.38	21.11	33.7	65.2	52.2	66.0	35.4	50.5
NormFormer	124.47	**20.24**	34.9	67.1	52.3	66.3	38.0	**51.7**
GPT3 (paper)	355.00	-	43.6	70.2	52.1	68.5	43.2	55.5
GPT3 (replicated)	354.74	15.41	45.7	71.1	53.4	72.3	40.8	56.7
NormFormer	354.98	**14.90**	47.6	71.0	54.1	71.9	41.8	**57.3**

Table 1 zero shot accuracy for causal LMs on a subset of the tasks evaluated by GPT3, using the prompts proposed in Brown et al. (2020). NormFormer outperforms the baseline at both the 125M parameter and 355M parameter sizes. Table 2 shows fine-tuned accuracy for MLMs on the GLUE tasks.

4 Analysis

We observe a mismatch in the magnitude of the updates to fully connected parameters in the baseline, similar to that which Pre-LN was introduced to alleviate. Figure 3 shows the average L1 norm of the

Table 2: Masked LM: Pre-training validation perplexity (PPL) and fine-tuned performance on GLUE Tasks for equally trained roberta-base variants.

	Model Size	PPL	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	Avg
Baseline	125.42	3.42	74.3	85.9	84.6	91.6	90.7	66.4	92.9	83.8
NormFormer	125.50	**3.32**	82.6	86.3	86.0	91.9	91.3	67.9	93.8	**85.7**

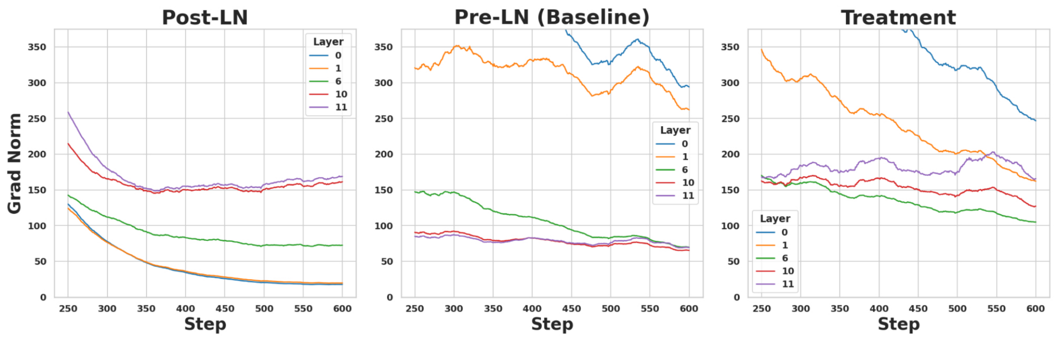

Figure 3: Average L1 Norm of gradient to second fully connected weight for layers 0,1,6 and 11, early in training.

gradient for second fully connected weight in various layers. Each line represents a different layer in a 12 layer, 125M parameter CLM and each sub plot shows a different architecture. For Post-LN, the gradients to later layers are large and the gradients to early layers vanish. Pre-LN changes this, increasing the grad norm to early layers. The introduction of more normalization, most notably LN between fully connected layers alleviates this problem, bringing the average norm for the layers closer together. Figure 4 shows the distribution of the γ parameters introduced in all three operations.

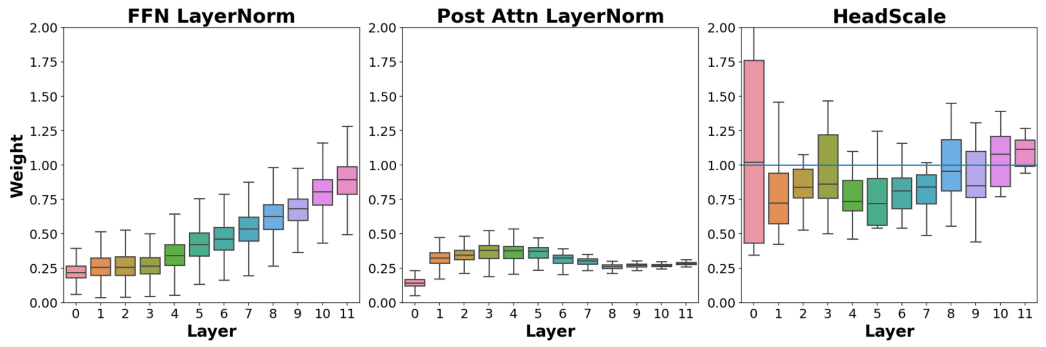

Figure 4: Distribution of γ parameters in each added operation. For FFN LN, earlier layers receive downscaled inputs, keeping their gradients in the same range as the gradients of later layers. This plot is discussed in detail in Section 4.

For the FFN LN, the parameters are smaller for earlier layers, reducing the magnitude of the inputs to early fully connected parameters, and thereby decreasing the magnitude of their gradients. The pattern for the post attention LN, in the middle of Figure 4, is only visible at layer 0, but all layers have coefficients below 1, indicating downscaling.[1] The `HeadScale` parameters, shown in the rightmost plot in Figure 4 vary more than the others, and have no relationship with depth in the network. We interpret this as evidence that the `HeadScale` parameters dynamically increase the importance of well initialized attention heads, as suggested in Chen et al. (2021).

One result of reducing the gradient mismatch, besides better perplexities and downstream task performance, is the ability to train at higher learning rates without loss explosion. To measure the stability of an architecture, we train it on a learning rate schedule with a very large peak learning rate and very long warmup, so that the learning rate increases a little each step until the loss explodes. Table 3 shows that NormFormer models last 150 steps longer and reach a higher peak learning rate than the baseline. For the baseline, loss explodes because the activations produced by multiplying the query and key features at layer 0 are too large to be represented in 16 bits. The down scaling of the attention outputs allows NormFormer to avoid this issue and last a bit longer.

[1]The downscaling is more apparent in Figure 5 in the Appendix, which plots the change in grad norm for each operation at each layer. It shows that adding extra normalization reduces the gradient norm for all attention parameters at every layer and that only FFN parameters at later layers, have increased gradient norms.

	Baseline	NormFormer
Peak LR	0.02	**0.0275**
Last Step	400	**550**
Failed Operation	$q@k$	Output Projection
Failed Layer	0	12

Table 3: LR Stability Test

5 Ablations

This section provides evidence that removing any of our additions to the transformer block degrades performance on language modeling tasks, and that our additions improve language modeling performance across a wide range of hyperparameter settings and model sizes. Experiments are run with the default hyperparameters in Table 4 for 470 V100 Hours (100,000 updates for the baseline) unless otherwise mentioned.

Learning Rate	0.003
Batch Size	524K Tokens
Parameters	124M+
Layers	12
Layer Dimension	768
Dropout	0
LR Warmup Updates	500
LR Scheduler	Linear Decay
Sequence Length	1024
Train Budget	470 V100 Hours

Table 4: Hyperparameters for ablation runs. This train budget allows the baseline model to run for 100,000 updates.

Scaling More Or Less Frequently Degrades Performance Table 5 shows that none of the three introduced operations can be removed without degrading performance. Rows 2, 3 and 4 remove each operation one at a time. In all cases perplexity increases, with the removal of `HeadScale` being the most damaging. In Row 5, ”+ 3 LN” we try to introduce more normalization inside self attention, applying LN to the query, key and value features in addition to our 3 other operations, for a total of 6 new operations. In this setting, every non LN operation inside the transformer layer is followed by an LN. We find that this does not change perplexities at a fixed number of updates, but reduces training speed by another 5%. This result suggests that there is not much upside to adding even more normalization after our three additions.

Architecture	Valid PPL
NormFormer	**15.98**
- Post-Attn LN	16.04
- FFN LN	16.19
- Head Scale	16.31
+ 3 More LN	15.99
Baseline	16.37

Table 5: Language Modeling Validation perplexities after 470 V100 Hours of pre-training. Removing any of our proposed additions degrades performance (Rows 2-4). Adding more normalization inside the Multi Headed Attention (Row 5). Does not impact perplexity at a fixed number of updates, but reduces throughput such that the model can only complete 87,500 updates vs. 92,500 for Rows 1-4 and 100,000 for Row 6.

Model Size We see similar improvement from the addition of extra normalization as we scale model size. At 125M parameters (row 5 of Table 6) , we show a run where we add attention and fully connected parameters instead of LN parameters by increasing the hidden dimension from 768 to 780. This result shows that the improvement from extra normalization is a more efficient way to add parameters, outpacing normal scaling laws.

	Parameters	Num Updates	Valid PPL	HParams
NormFormer	124,471,296	475,000	**14.87**	-
Baseline	124,379,136	520,000	15.18	-
NormFormer	124,425,216	92,500	**15.92**	GPT3
Baseline	124,379,136	100,000	16.36	GPT3
Baseline	127,670,400	85,000	16.29	GPT3
NormFormer	354,988,032	177,000	**11.85**	-
Baseline	354,742,272	200,000	12.04	-
NormFormer	1,313,710,080	85,000	**10.25**	-
Baseline	1,313,460,224	95,000	10.48	-

Table 6: Varied Sizes. Groups separated by vertical lines have comparable training budgets.

Other Hyperparameter Settings Table 7 shows language modeling perplexities for 7 different hyperparameter configurations, separated by horizontal lines. `NormFormer` outperforms the baseline in all settings.

	Learning Rate	Setting Changes	Valid PPL
Baseline	0.001	-	16.80
NormFormer	0.001	-	**16.43**
Baseline	0.003	-	16.37
NormFormer	0.003	-	**15.98**
Baseline	0.006	-	16.58
NormFormer	0.006	-	**16.26**
Baseline	0.003	Longer Warmup	16.50
NormFormer	0.003	Longer Warmup	**16.12**
Baseline	0.003	GPT3	16.29
NormFormer	0.003	GPT3	**15.92**
Baseline	0.003	Clip Grad Norms at 0.1	16.46
NormFormer	0.003	Clip Grad Norms at 0.1	**16.20**

Table 7: Longer Warmup: increase LR Warmup to 6,000 steps (from 500). GPT3: increase sequence length to 2048, increase dropout to 0.1, increase training budget to 1,000 V100 hours. Grad Clip: clip gradient norms at 0.1. NormFormer outperforms the baseline in all settings.

6 Related Work

Xiong et al. (2020) shows that for Post-LN: gradients are too big for later layers and solves this problem with Pre-LN. We build on the Pre-LN architecture to make it even more stable and efficient.

Press et al. (2020a) proposes an architecture where instead of interleaving attention and feed forward sublayers, the attention all happens first. This increases the number of late FFN parameters, rather than increasing their importance and gradient norm, as our FFN LN does, and does not impact stability.

Our `HeadScale` operation is similar to Chen et al. (2021), but used differently. Whereas that work prunes attention heads with low γ parameters, we use the γ parameters to improve pretraining performance.

Replacing the FFN LN with the FFNGeGlu proposed in Shazeer (2020), which includes scaling but no normalization, degraded performance in our setting.

We also find that the LN variant proposed in Raffel et al. (2020), which removes the bias and the mean substraction from the normalization, performs equally well to our LN and has fewer trainable parameters, but is about 2x slower than the `FusedLayerNorm` implementation we use.

7 Conclusion

We identify a mismatch in the gradients of Pre-LN transformer weights: earlier layers receive much larger gradients than later layers. We propose `NormFormer`, which alleviates the mismatch by adding 3 extra normalization operations to each transformer layer These modifications alleviate the gradient mismatch for fully connected parameters and improve validation perplexity and downstream task performance for both causal and masked language models. None can be removed without degrading performance back towards the baseline, and adding more normalization does not improve performance. Since NormFormer primarily addresses the gradient mismatch by increasing the gradients to the last FFN layers while decreasing the gradient magnitudes in other parts of the network, future work could examine whether all 3 operations need to be added to every layer. Additionally, the small computational increase associated with NormFormer could be alleviated by fusing the FFN LN with the preceding fully connected layer, with or without the mean centering and bias, which do not appear to improve pre-training perplexity.

8 Appendix

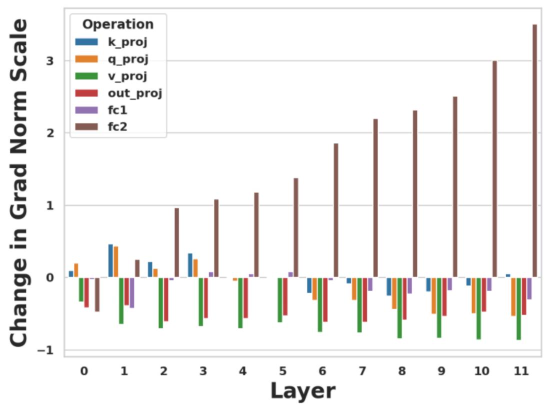

Figure 5: Change in grad scale norm for each operation of NormFormer compared to the baseline. Norms are the average between step 950 and 1000, normalized to control for different losses. 2.0 on the Y axis means the gradient to a parameter is twice as large, on average. The NormFormer increases the norm to fully connected parameters in later layers, while reducing the gradient norm to attention parameters at all layers. The results are discussed in detail in Section 4.

Wikitext103 Table 8 shows that NormFormer can also provide gains on top of a well tuned language model with much less data. We simply add our three normalization steps to the architecture and hyperparameters of Baevski & Auli (2019), and find that (a) convergence perplexity improves modestly, (b) the normalized model does not learn anything in the last 20% of steps, indicating that if hyperparameters were re-tuned NormFormer might outperform the baseline by more.

	Steps to Final PPL	PPL
Baseline	100%	18.70
NormFormer	80%	18.65

Table 8: Wikitext 103 results following Baevski & Auli (2019). `Steps to Final PPL`: at what percentage of the 280K steps did the model reach its final perplexity. `PPL`: Best Perplexity

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=ByxZX20qFQ`.

Yonatan Bisk, Rowan Zellers, Ronan Le bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439, Apr. 2020. doi: 10.1609/aaai.v34i05.6239. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6239`.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. Earlybert: Efficient bert training via early-bird lottery tickets, 2021.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. 2020.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. Jurassic-1: Technical details and evaluation. Technical report, AI21 Labs, August 2021.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. FAIRSEQ: A fast, extensible toolkit for sequence modeling. 2019.

Ofir Press, Noah A. Smith, and Omer Levy. Improving transformer models by reordering their sublayers, 2020a.

Ofir Press, Noah A Smith, and Mike Lewis. Shortformer: Better language modeling using shorter inputs. *arXiv preprint arXiv:2012.15832*, 2020b.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. 21:1–67, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, Apr. 2020. doi: 10.1609/aaai.v34i05.6399. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6399`.

Noam Shazeer. Glu variants improve transformer, 2020.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. 2019.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *arXiv preprint arXiv:1506.06724*, 2019.