BTLM-3B-8K: 7B Parameter Performance in a 3B Parameter Model

Nolan Dey[*1], **Daria Soboleva**[*1], **Faisal Al-Khateeb**[1], **Bowen Yang**[1], **Ribhu Pathria**[1]

Hemant Khachane[1], **Shaheer Muhammad**[1], **Zhiming (Charles) Chen**[1], **Robert Myers**[2]

Jacob Robert Steeves[2], **Natalia Vassilieva**[1], **Marvin Tom**[1], **Joel Hestness**[1]

[1] Cerebras Systems, [2] OpenTensor Foundation

{nolan,daria.soboleva}@cerebras.net

Abstract

We study recent techniques targeted to improve the parameter efficiency and modeling quality of large language models (LLMs). We experiment with recently-proposed training approaches, such as overtraining for a large number of tokens-per-parameter on a high-quality dataset, carefully tuning hyperparameters with maximal update parameterization (μP), and adjusting learning rate and batch size. We also test recent state-of-the-art model features, namely, rotary and ALiBi position embeddings, and the Swish-gated linear unit (SwiGLU). We find a pretraining recipe that improves over Cerebras-GPT μP validation loss by 12.7% for the same parameter budget.

With this recipe, we train the state-of-the-art 3B parameter foundation model, called the Bittensor Language Model ("BTLM-3B-8K"), which is sized to deploy easily on memory or compute-constrained devices. Over a broad set of downstream tasks, BTLM beats all other 3B foundation models by 2-5.5%, making it competitive with some 7B parameter models that are 2.5× larger. BTLM-3B-8K is available under an Apache 2.0 license on Hugging Face: `https://huggingface.co/cerebras/btlm-3b-8k-base`.

1 Introduction

Large Language Models (LLMs) can perform a diverse collection of text-based tasks with brief instructions [2], making them useful in many settings. Models with at least 7 billion parameters are considered to provide strong language and reasoning capabilities. Unfortunately, 7B models are hard to deploy for inference on devices with memory or compute limitation, because even when quantized, they require 6-9GB of memory, almost double that of a device like the iPhone 13 (4GB).

We aim to develop a state-of-the-art 3B model capable of inference on memory- and compute-constrained devices. We leverage open-source scaling laws, starting from Cerebras-GPT [9], which provide compute-optimal projections for LLM model quality. By training small models, we ablate the effects of recent LLM techniques to improve parameter efficiency and model quality.

[*]Equal contribution

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

In this process, we identify a new LLM training recipe that improves over Cerebras-GPT μP performance by 12.7%, improving parameter efficiency. We test training algorithm advancements such as overtraining on many tokens-per-parameter [14, 33] and maximal update parameterization to improve training dynamics and hyperparameter tuning [35]. We also test model architecture changes that improve model quality. Specifically, we test rotary (RoPE) [29] and ALiBi position embeddings [24], and the Swish-gated linear unit (SwiGLU) [27]. Finally, we identify a better learning rate decay schedule that improves loss when training on a large number of tokens per parameter.

We use our new recipe to train the state-of-the-art 3B parameter foundation model, called the Bittensor Language Model ("BTLM-3B-8K"). BTLM-3B-8K is sized to deploy easily on memory- or compute-constrained devices, such as laptops and mobile phones. Over a broad set of downstream tasks, BTLM beats all other 3B foundation models by 2-5.5%, and it is competitive with some 7B parameter models, which require 2.5× more inference memory and compute.

2 Model Architecture and Training Experiments

In this section, we experiment with state-of-the-art training approaches and architectural changes and measure how they affect training efficiency. Overtraining on many tokens-per-parameter improves loss by 7.7% for the same parameter budget, while roughly even loss improvement can be attributed to training algorithm adjustments (2.8%) and model architecture features (2.7%). Overall the combination of these features gives a 12.7% improvement in validation loss over the Cerebras-GPT μP baseline.

Baseline Model Our baseline model is Cerebras-GPT μP with 111M parameters, trained with 20 tokens per parameter ("TPP") on The Pile dataset [10] at sequence length of 2,048 tokens. Table 1 lists the model and training hyperparameters for our baseline model ("Cerebras-GPT"), the features we experiment with, and our final selected training recipe.

Table 1: Overview of baseline, experimentally tested features, and final BTLM recipe.

Features	**Cerebras-GPT**	**Experiments**	**BTLM Recipe**
Model Dimensions	$d_{\text{model}} = 768$	$n_{\text{layers}} = 10$	$d_{\text{head}} = 64$
Activation Function	GeLU	SwiGLU	SwiGLU
Position Embeddings	Learned	RoPE, ALiBi	ALiBi
Batch Size (Sequences)	120	120 – 420	420
LR Decay Ratio (r_{decay})	10×	10× – 370×	118× (TPP/2)
μP Hyperparameters			
Proxy model's layer width ($d_{\text{model,base}}$)	256	256	256
Base Learning Rate (η_{base})	6e-3	8e-3 – 2e-2	1.2e-2
Weight Initialization Std. Dev. (σ_{base})	0.08	0.07 – 0.12	0.073
Embedding Multiplier (m_{emb})	10.0	1.0 – 20.0	14.6
Output Logits Multiplier (m_{out})	1.0	0.35 – 3.0	2.22

2.1 Model Architecture Improvements

SwiGLU We replace the GeLU non-linearity with the SwiGLU activation function [27]. To ensure similar compute FLOPs to the GeLU model, which uses feed-forward network filter size $d_{\text{ffn}} = 4d_{\text{model}}$, we adjust the SwiGLU filter size to $d_{\text{ffn}} = \frac{8}{3}d_{\text{model}}$ to account for the additional projection.

ALiBi and RoPE We experiment with both ALiBi [24] and RoPE [4] positional embeddings. Unlike learned embeddings, these methods inject position information into the model at every layer, and not just at the initial one, which leads to performance gains. Here, we focus exclusively on the performance observed at a context length of 2,048.

2.2 Training improvements

Overtraining on many Tokens-Per-Parameter We follow prior studies that show training with higher TPP can improve model quality for a given parameter count. In particular, we aim to train a 2.6B paramter model on SlimPajama (627B tokens or 236.4 TPP), so we choose that TPP to study at the small scale.

LR Decay Ratio We execute a line search to determine the optimal learning rate decay ratio, discovering a heuristic value that approximates TPP/2, which aligns with a 10x decay when set at 20 TPP. Comprehensive details are available in Appendix A. The experimental values for r_{decay} can be found in Table 1.

µP We follow [9] µP hyperparameters (HPs) search experiments explained in the Appendix G.2. In our experiments we found its beneficial to use different µP parameters as outlined in the BTLM Recipe column in Table 1.

2.3 Ablation Results and Discussion

We compare modeling and training techniques by looking at the loss improvement in the Table 2 and the trade-offs between loss improvements versus increase in the parameter counts in the Figure 1. Additionally, a plot illustrating FLOP budget against loss can be found in Appendix C for further reference. Overtraining to 236 TPP gives a big parameter-efficiency boost, improving loss by 7.7% at the expense of 11.8x more FLOPs. According to Cerebras-GPT scaling laws, this is FLOP-inefficient by 8.2%, because it is well past the TPP point of diminishing returns. But it is a big overall gain that we deem to be worth it for compactness of the model. The next biggest gains come from tuning the training hyperparameters (muP and learning rate schedule), giving 2.8% gain. Finally, model architecture features, SwiGLU and ALiBi (or RoPE) show 2.7% gain when combined.

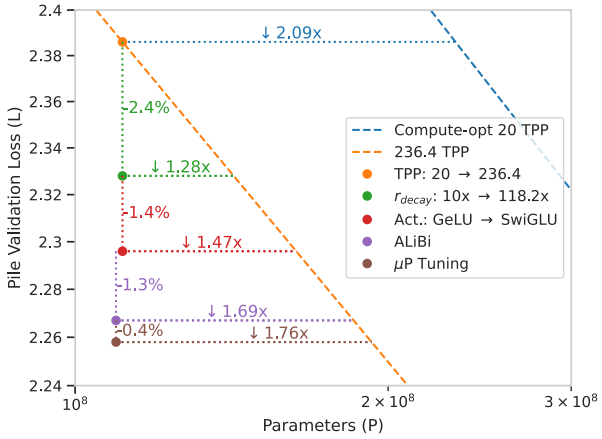

Figure 1: Cross-entropy loss at specific parameter counts, highlighting how modifying tested features contributes to improved parameter efficiency.

Table 2: Ablation results showing loss improvements and changes in training FLOPs for each ablation starting from the Cerebras-GPT µP, 111M baseline.

Variant	Loss	FLOPs
Baseline: Cerebras-GPT µP, 111M	2.586	2.23e18
TPP: 20 → 236	2.386	2.63e19
r_{decay} : 10× → 118×	2.328	2.63e19
Act.: GeLU → SwiGLU	2.296	2.63e19
↳ RoPE	2.259	2.60e19
↳ ALiBi	2.267	2.60e19
↳ µP Tuning	2.258	2.60e19

3 Scaling to BTLM-3B-8K

In this section, we specify the architecture and training configuration for BTLM-3B-8K. We present downstream task results, which show that BTLM beats all other 3B foundation models by 2-5.5% and is even competitive with some 7B parameter models.

3.1 BTLM Architecture and Training Configuration

Architecture BTLM-3B-8K is a GPT-3-like autoregressive transformer decoder model with dense multi-head attention. Based on experiments in Section 2.2, we incorporate three changes into the architecture. First, BTLM uses µP, adding scaling factors to control activation magnitudes and improve training dynamics. Further, we use the SwiGLU nonlinearity and ALiBi position embeddings.

Hyperparameters BTLM-3B-8K uses common dimensions for a "3B" parameter model (actual parameter count is 2.6B). Hidden size is 2560, and it contains 32 decoder blocks. The multi-head attention contains 80 heads, and the feed-forward networks have filter size 6826. We train with the best µP hyperparameters from our sweep (see µP parameters in "BTLM Recipe" column in Table 1).

Table 3: Average accuracy across groups of downstream tasks. All tasks use 0-shot evaluation, except MMLU which is 5-shot.

Model		Pre-training (↓)		Downstream task accuracy (% ↑)				
		Tokens	FLOPs	CSR	WK	RC	MMLU	Code
StableLM-Alpha-v2 [34]	2.7B	1.1T	2.1e22	58.0	31.7	48.1	26.6	9.7
RedPajama-INCITE [32]	2.6B	800B	1.5e22	56.7	34.6	48.4	27.0	5.0
OpenLLaMA 3B v2 [12]	3.3B	1.0T	2.2e22	57.7	33.7	47.7	26.6	9.5
BTLM-3B-8K (Ours)	2.6B	**627B**	**1.3e22**	**59.9**	**36.6**	**50.0**	**28.1**	**9.9**
RedPajama-INCITE [32]	6.7B	1.0T	4.4e22	59.5	40.1	50.0	27.5	5.2
OpenLLaMA [12]	6.6B	1.0T	4.3e22	58.6	41.7	50.2	30.1	7.7
Mosaic MPT [31, 30]	6.7B	1.0T	4.4e22	63.2	42.7	50.7	28.5	**15.4**
LLaMA [33]	6.6B	1.0T	4.3e22	**63.7**	**45.3**	**52.1**	**35.2**	12.1

Training Dataset We train BTLM on a high quality dataset, SlimPajama [28] (Apache 2.0) (a deduplicated version of RedPajama [8]), which contains 627B tokens (236 TPP for a 2.6B parameter model). To ensure BTLM's ability to perform inference on long context lengths, we train the model in two phases: 470B tokens with a context length of 2,048 and 157B tokens with 8,192.

Optimizer, Training Schedule We train with the AdamW optimizer using hyperparameters $\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 10^{-8}$, weight decay of 0.1 and gradient clipping to a maximum norm of 1.0. We warm the learning rate to its maximum value of 1.2e-2 over 375M tokens, and then linearly decay by a factor of 118.2× to 0.85% of the base.

3.2 BTLM-3B-8K Evaluation

We evaluate models on zero-shot and few-shot downstream tasks using the Eleuther LM evaluation harness ([11]) in the Table 3[2]. We compare BTLM-3B-8K with many 3B and 7B parameter open-source foundation models on a broad set of task domains: common sense reasoning (CSR) [1, 26, 36, 25, 21], world knowledge (WK) [7, 17, 16], reading comprehension (RC) [18, 6], massive multitask language understanding (MMLU) [13], and coding [3].

BTLM-3B-8K achieves state-of-the-art performance among 3B parameter models, outperforming others by a substantial margin while using the least pretraining compute and data. BTLM-3B-8K was trained on 627B tokens, significantly less than RedPajama-INCITE-3B at 800B tokens and OpenLLaMA 3Bv2 at 1T tokens. BTLM-3B is even competitive with 7B models, outperforming RedPajama-INCITE-7B [32], OpenLLaMA-7B [12], and StableLM-Alpha-7B-v2 [34] in various task domains, despite using 3.3x less training compute and 1.6x less training data. We believe that key contributors to BTLM's performance are the high-quality, deduplicated SlimPajama corpus, tuned μP hyperparameters,optimized r_{decay}, SwiGLU nonlinearity, and ALiBi position embeddings.

We do not perform an ablation to quantify the efficiency benefits of training on deduplicated data. Also even greater parameter efficiency could have been achieved by training for more than 236.4 TPP.

4 Conclusion

We comprehensively study recent large language model techniques to improve parameter efficiency and modeling quality. In small-scale experiments, we find a training recipe that improves over Cerebras-GPT μP validation loss by 12.7% at the same parameter budget. This improvement can be attributed to overtraining on many Tokens-Per-Parameter (7.7%), training algorithm adjustments (2.8%) and model architecture features (2.7%). We demonstrate the parameter efficiency improvement from this recipe by training the Bittensor Language Model ("BTLM-3B-8K"), a 3B parameter model sized to deploy easily on memory or compute-constrained devices. Over a broad set of downstream tasks, BTLM beats all other 3B foundation models by 2-5.5%, making it competitive with some 7B parameter models that are 2.5× larger.

[2]Full results are listed in Appendix E

References

[1] 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*, 2020.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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, 2020.

[3] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating Large Language Models Trained on Code, 2021.

[4] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending Context Window of Large Language Models via Positional Interpolation, 2023.

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling Language Modeling with Pathways. 2022.

[6] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions, 2019.

[7] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge, 2018.

[8] Together Computer. RedPajama: An Open Source Recipe to Reproduce LLaMA training dataset, 2023.

[9] Nolan Dey, Gurpreet Gosal, Zhiming, Chen, Hemant Khachane, William Marshall, Ribhu Pathria, Marvin Tom, and Joel Hestness. Cerebras-GPT: Open Compute-Optimal Language Models Trained on the Cerebras Wafer-Scale Cluster, 2023.

[10] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800GB Dataset of Diverse Text for Language Modeling, 2020.

[11] Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 2021.

[12] Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, 2023.

[13] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding, 2020.

[14] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack William Rae, and Laurent Sifre. An Empirical Analysis of Compute-optimal Large Language Model Training. In *The Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[15] Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient Attentions for Long Document Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[16] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension, 2017.

[17] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 2019.

[18] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding Comprehension Dataset From Examinations, 2017.

[19] Dacheng Li*, Rulin Shao*, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. How Long Can Open-Source LLMs Truly Promise on Context Length?, 2023.

[20] Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An Empirical Model of Large-Batch Training, 2018.

[21] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[22] OpenAI. Gpt-4 technical report, 2023.

[23] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only, 2023.

[24] Ofir Press, Noah A. Smith, and Mike Lewis. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation, 2021.

[25] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. *Communications of the ACM*, 2021.

[26] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. SocialIQA: Commonsense Reasoning about Social Interactions, 2019.

[27] Noam Shazeer. GLU Variants Improve Transformer, 2020.

[28] Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://www.cerebras.net/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama, 2023.

[29] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding, 2022.

[30] MosaicML NLP Team. Announcing MPT-7B-8K: 8K Context Length for Document Understanding, 2023.

[31] MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms, 2023.

[32] Together.ai. Redpajama Models V1, 2023.

[33] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, 2023.

[34] Jonathan Tow. StableLM Alpha v2 Models, 2023. Additional model available at: https://huggingface.co/stabilityai/stablelm-base-alpha-3b-v2.

[35] Greg Yang, Edward Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer. In *Advances in Neural Information Processing Systems*, 2021.

[36] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence?, 2019.

[37] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

[38] Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. QMSum: A New Benchmark for Query-based Multi-domain Meeting Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

Appendix

A Learning Rate Decay Sweep

To test the learning rate decay ratio scaling heuristic of r_{decay} = TPP/2 presented in Section 2.2, we sweep the learning rate decay ratio (r_{decay}) for a 111M model trained on the Pile dataset. In Figure 2 we find that r_{decay} larger than TPP/2 improves Pile validation loss for both 370 TPP large batch size and 236.4 TPP small batch size setting, suggesting this heuristic is useful. Furthermore, [14] found $r_{decay} = 10$ to be optimal for 20 TPP models, also coinciding with our TPP/2 heuristic.

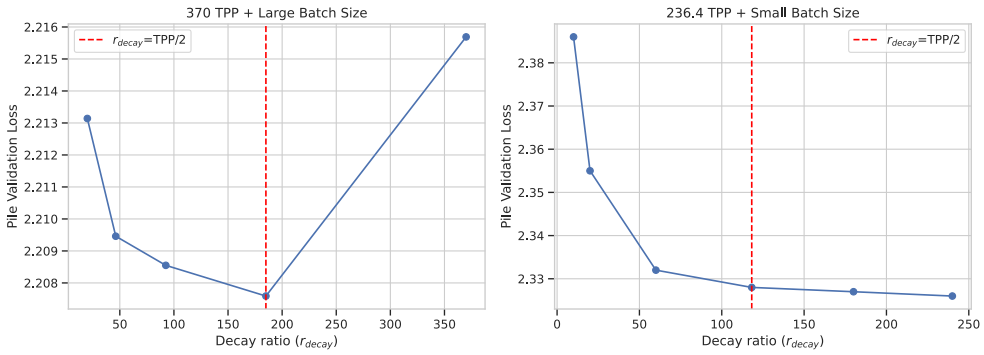

Figure 2: Shows greater than 20tpp 10x proportional could be best. But this is in a small BS setting relative to the final setup.

B µP Hyperparameters Search Details

Considering the substantial impact of maximal update parameterization (µP) on the stability during training, precision of scaling law predictions, and the quality of the model, we have integrated it into our experimental framework. We follow [9] µP hyperparameters (HPs) search experiments explained in the Appendix G.2. Transferring the learning rate can be less effective when the proxy model is trained with a batch size smaller than the critical batch size, especially when the larger model is trained at or above this critical batch size [20]. To mitigate this sub-optimality weincorporate a larger batch size in this experiment to optimize learning rate transfer. We provide the full list of µP parameters as well as their experimental ranges in the Table 1.

C FLOP efficiency

Figure 3 demonstrates cross-entropy loss for different modeling and architectural changes and their corresponding loss improvements at different FLOP budgets.

D Training Loss Stability

It is common for LLMs to encounter loss instability which can lead to loss divergence and require careful manual interventions to recover training ([37, 5]). Figure 4 shows that BTLM training progressed with excellent loss stability, especially given how large our learning rate is relative to other models. We attribute this stability to the maximal update parameterization which controls activation explosion as model width is scaled up. BTLM only experienced two loss spikes: one at step 15 (59M tokens) and another at the transition to 8,192 context length training as the model adapts to longer sequences. The training fully recovered from both spikes, and they did not seem to impact the overall trajectory of the loss curve.

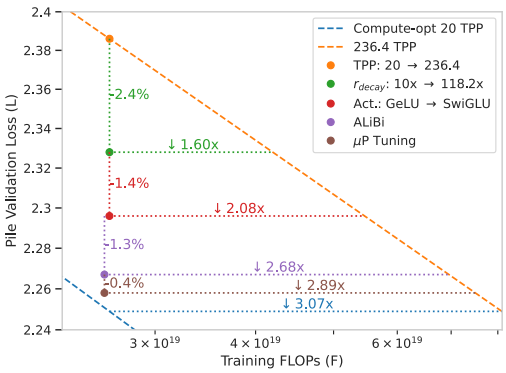

Figure 3: Cross-entropy loss at specific FLOP counts. We demonstrate how improving parameter efficiency might negatively effect compute efficiency, and needs to be studied together.

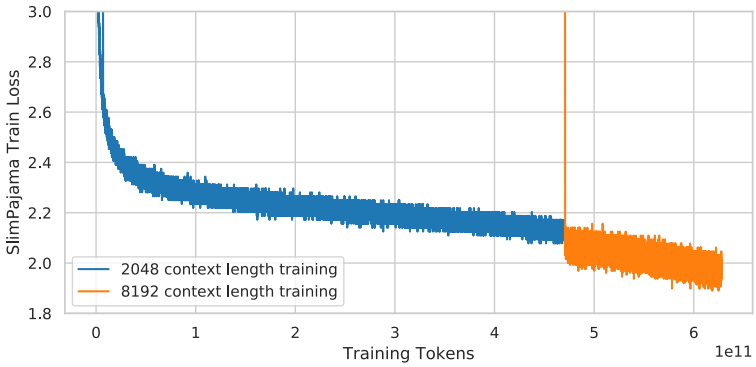

Figure 4: SlimPajama train cross-entropy loss versus training tokens. We train for 470B tokens with sequence length of 2,048 and 157B tokens with sequence length of 8,192.

E Full Downstream Evaluation Results

We demonstrate full downstream evaluation results on the standard tasks in the Table 4 with average results for individual groups such as common sense reasoning (CSR), world knowledge (WK), reading comprehension (RC), massive multitask language understanding (MMLU), and code tasks. Tables 5, 6, 7 contain detailed results for each task in the grouped categories. We provide results on long-context benchmarks [15, 38, 19] in the Table 8 and on the Figure 5. Finally, we evaluate BTLM model on the safety benchmarks and outlines results in the Table 9. Overall BTLM exhibits bias, toxicity, and truthfulness like existing models. Nevertheless, we recommend exploring harm mitigation strategies in deployment contexts [22]. Additionally, more careful dataset curation techniques such as filtering not-safe-for-work URLs [23] showed to be helpful in reducing model harmfulness.

F BTLM-3B-8K SlimPajama Extrapolation

To further assess BTLM's extrapolation capability, we evaluate on the SlimPajama test set with 32,768 context length and plot loss at each token position in Figure 6. We evaluate checkpoints at different points during training to gain insight into how extrapolation capability evolves.

[24] report that ALiBi grants impressive extrapolation properties with a 255M parameter model trained on 103M tokens. This corresponds to just 0.4 tokens TPP, well below the 20 TPP recommendation from [14]. With our 1.2 TPP checkpoint we observe similar extrapolation performance as [24] but this result appears to only be possible due the overall loss being quite poor quite early in training. As training progresses, our model learns to "overfit" to the current context length. We observe that the final checkpoint from the 2,048 context length training phase (75% complete) cannot extrapolate

Table 4: Average accuracy on common sense reasoning (CSR), world knowledge (WK), reading comprehension (RC), massive multitask language understanding (MMLU), and code tasks. All tasks are using 0-shot evaluation, except MMLU which is 5-shot. Code accuracy refers to HumanEval pass@1 accuracy.

Model		Pre-training (↓)		Downstream task accuracy (↑)				
		Tokens	FLOPs	CSR	WK	RC	MMLU	Code
StableLM-Alpha-3B-v2	2.7B	1.1T	2.10e22	58.0	31.7	48.1	26.6	9.7
RedPajama-INCITE-3B	**2.6B**	800B	1.50e22	56.7	34.6	48.4	27.0	5.0
OpenLLaMA 3Bv2	3.3B	1T	2.20e22	57.7	33.7	47.7	26.6	9.5
BTLM-3B-8K	**2.6B**	**627B**	**1.3e22**	**59.9**	**36.6**	**50.0**	**28.1**	**9.9**
StableLM-Alpha-7B-v2	6.7B	1.1T	4.90e22	61.2	38.3	48.1	26.6	15.0
Falcon-7B	6.9B	1.5T	7.00e22	63.4	45.0	51.1	26.3	0.0
RedPajama-INCITE-7B	6.7B	1T	4.40e22	59.5	40.1	50	27.5	5.2
Falcon-RW-7B	**6.3B**	**350B**	**1.5e22**	61.0	39.1	49.8	26.2	N/A
OpenLLaMA 7B	6.6B	1T	4.30e22	58.6	41.7	50.2	30.1	7.7
MPT-7B	6.7B	1T	4.40e22	63.2	42.7	50.7	28.5	**15.4**
XGen-7B-8K	6.7B	1.5T	7.10e22	60.7	40.0	51.5	35.9	14.2
OpenLLaMA 7Bv2	6.6B	1T	4.30e22	60.5	40.7	50.7	40.4	14.7
LLaMA-7B	6.6B	1T	4.30e22	**63.7**	45.3	52.1	35.2	12.1
LLaMA-2-7B	6.6B	2T	9.30e22	63.4	**47.5**	**53.2**	45.8	13.7

Table 5: Zero-shot validation accuracy on each common sense reasoning task, except for OpenBookQA which uses the test split.

Model	Common Sense Reasoning (↑)					
	PIQA	SIQA	HellaSwag	WinoGrande	OBQA	Avg.
RedPajama-INCITE-Base-3B-v1	73.8	44.9	63.2	63.6	37.8	56.7
OpenLLaMA 3Bv2	76.2	44.8	65.2	63.3	39.2	57.7
StableLM-Base-Alpha-3B-v2	**77.2**	44.1	65.8	62.3	**40.8**	58.0
BTLM-3B-8K	**77.2**	**46.5**	**69.8**	**65.8**	40.4	**59.9**
OpenLLaMA 7B	74.5	46.9	64.7	66.8	40.0	58.6
RedPajama-INCITE-7B-Base	77.4	45.1	70.4	64.0	40.4	59.5
OpenLLaMA 7Bv2	78.2	47.0	69.6	65.8	42.0	60.5
XGen-7B-8K-Base	75.9	47.9	74.2	65.5	40.2	60.7
Falcon-RW-7B	79.1	46.6	72.1	65.7	41.4	61.0
StableLM-Base-Alpha-7B-v2	79.8	44.1	71.7	69.1	41.2	61.2
MPT-7B	**80.6**	48.1	76.2	68.1	42.8	63.2
Falcon-7B	80.5	**49.1**	**76.3**	67.1	44.	63.4
LLaMA-2-7B	79.0	49.0	76.0	68.9	44.2	63.4
LLaMA-7B	79.2	48.5	76.2	**70.0**	**44.4**	**63.7**

well beyond 2,048 context length. This demonstrates that ALiBi alone does not provide competitive extrapolation capability, and we suggest using variable context length training schedules to improve performance. The final BTLM-3B-8K model trained on 8,192 with a context length can extrapolate well up to ≈9,216 context length but suffers loss degradation beyond this.

Table 6: Zero-shot accuracy on reading comprehension and world knowledge tasks. We report test accuracy except for BoolQ, where we report validation accuracy.

Model	Reading Comprehension (↑)				World Knowledge (↑)				
	R-m	R-h	BoolQ	Avg.	ARC-e	ARC-c	NQ	TQA	Avg.
StableLM-Alpha-3B-v2	41.2	38.9	64.3	48.1	53.8	32.9	5.5	34.5	31.7
OpenLLaMA 3Bv2	40.6	36.8	65.6	47.7	61.9	35.1	6.3	31.5	33.7
RedPajama-INCITE-3B	40.1	37.9	67.4	48.5	61.6	34.4	6.4	**36.0**	34.6
BTLM-3B-8K	**40.6**	**39.4**	**70.0**	**50.0**	**66.9**	**37.6**	**6.9**	34.9	**36.6**
StableLM-Alpha-7B-v2	42.3	38.8	70.2	50.4	59.4	38.1	9.1	46.5	38.3
Falcon-RW-7B	41.7	38.6	69.1	49.8	67.9	38.7	9.8	39.9	39.1
RedPajama-INCITE-7B	41.2	38.2	70.8	50.1	69.3	39.2	5.5	46.2	40.1
OpenLLaMA 7B	42.3	37.7	70.5	50.2	67.1	37.1	12.2	50.3	41.7
MPT-7B	40.3	38.0	73.7	50.7	70.0	41.9	11.9	47.1	42.7
OpenLLaMA 7Bv2	41.2	38.7	72.3	50.7	68.0	40.2	7.9	46.9	40.7
Falcon-7B	42.3	37.2	73.8	51.1	70.8	43.5	**14.6**	50.9	45.0
XGen-7B-8K	41.2	39.0	74.2	51.5	66.9	41.1	07.2	44.6	40.0
LLaMA-7B	40.9	**40.3**	75.0	52.1	72.9	44.7	11.7	52.1	45.3
LLaMA-2-7B	**42.3**	39.5	**77.8**	**53.2**	**74.6**	**46.3**	12.5	**56.6**	**47.5**

Table 7: Five-shot accuracy on the Massive Multitask Language Understanding (MMLU) benchmark and zero-shot performance on HumanEval (HE) with pass@1 and pass@100 on the test splits.

Model	MMLU (↑)					Code (↑)	
	Hum.	STEM	Soc. Sci.	Other	Avg.	HE@1	HE@100
StableLM-Alpha-3B-v2	27.1	26.2	24.9	28.2	26.6	9.7	**33.3**
OpenLLaMA 3Bv2	25.7	26.0	26.6	28.5	26.7	9.5	32.9
RedPajama-INCITE-3B	26.2	26.6	29.6	25.9	27.1	5.0	13.3
BTLM-3B-8K	**27.6**	**27.1**	**27.9**	**29.8**	**28.1**	**9.9**	29.7
Falcon-RW-7B	27.3	23.2	25.6	27.7	26.0	N/A	N/A
Falcon-7B	26.9	25.9	24.4	27.6	26.2	0.0	1.8
RedPajama-INCITE-7B	26.2	27.4	30.6	26.4	27.7	5.2	19.2
MPT-7B	27.4	28.1	29.2	29.7	28.6	**15.4**	**54.2**
OpenLLaMA 7B	28.4	28.4	31.3	32.9	30.3	7.7	24.9
LLaMA-7B	34.0	30.6	38.4	38.2	35.3	12.1	35.9
XGen-7B-8K	33.6	29.8	39.5	41.6	36.1	14.2	41.5
OpenLLaMA 7Bv2	37.0	33.4	45.4	47.0	40.7	14.7	47.3
StableLM-Alpha-7B-v2	42.6	36.6	49.3	51.2	44.9	15.0	44.9
LLaMA-2-7B	**43.1**	**36.9**	**51.7**	**52.6**	**46.1**	13.7	43.6

Table 8: ROUGE scores on the QMSum and GovReports long text summarization tasks. To test the interpolation regime for models, we only evaluate samples less than 8,192 tokens in length.

Model		Pretraining (↓)		QMSum (↑)			GovReports (↑)		
		Tokens	FLOPs	R-1	R-2	R-L	R-1	R-2	R-L
XGen-7B-8K	6.7B	1.5T	7.0e22	11.8	3.0	9.1	11.8	5.6	8.3
MPT-7B-8K	6.7B	1.5T	7.1e22	14.8	**5.2**	11.3	8.5	3.9	6.2
BTLM-3B-8K	**2.7B**	**627B**	**1.3e22**	**16.3**	2.5	**12.4**	**15.5**	**5.8**	**10.2**

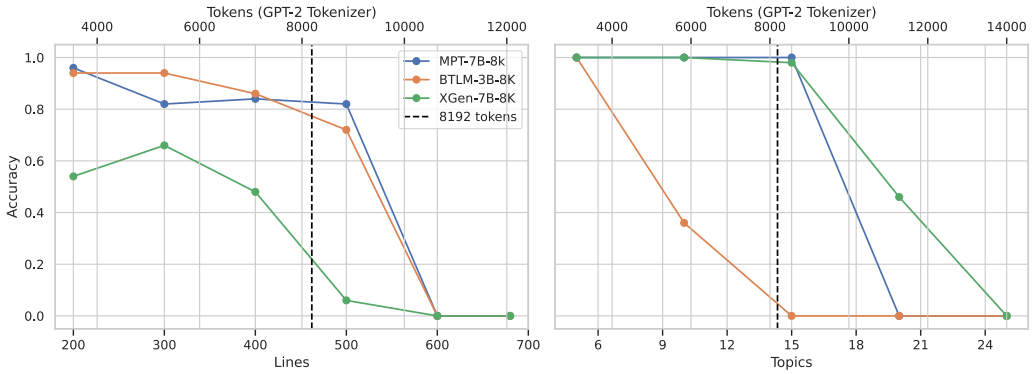

Figure 5: Accuracy on the LongEval-Lines and LongEval-Topics long-range retrieval tasks.

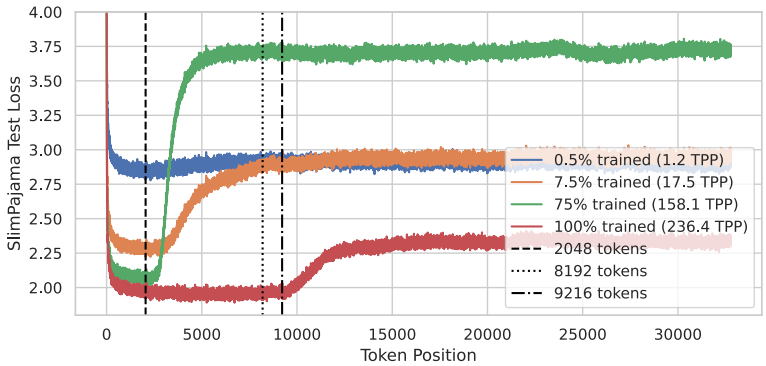

Figure 6: SlimPajama test set cross-entropy loss for various BTLM checkpoints at each token position. Inference is performed on examples packed to 32,768 tokens in length.

Table 9: Zero-shot evaluations on bias, toxicity, and truthfulness benchmarks: TruthfulQA, WinoGender, ToxiGen, and CrowS-Pairs.

Task	Subtask	BTLM-3B-8K	OpenLLaMA 3Bv2	RedPajama-INCITE-7B	Falcon-7B	LLaMA-2-7B
TruthfulQA ↑	Multiple choice	**35.9**	34.8	33.0	34.2	**39.0**
WinoGender ↑	hers/her/she	**60.0**	56.7	63.3	60.0	**69.2**
	his/him/he	**60.0**	56.7	60.0	55.0	**62.5**
	their/them/someone	57.5	**60.0**	**72.5**	56.7	69.2
	hers/her/she (gotcha)	**48.3**	37.9	48.3	41.4	**62.1**
	his/him/he (gotcha)	29.0	**35.5**	51.6	45.2	**67.7**
	All	**59.2**	57.8	65.3	57.2	**66.9**
ToxiGen ↓	Multiple choice	50.7	**44.6**	45.3	52.7	57.8
CrowS-Pairs ↓	Age	75.8	**53.9**	**71.4**	**71.4**	74.7
	Disability	69.2	**64.6**	76.9	**67.7**	**67.7**
	Gender	67.2	**53.8**	68.4	66.9	**62.5**
	Nationality	60.2	**52.3**	62.5	61.1	**59.7**
	Physical Appearance	77.8	**66.7**	79.2	76.4	77.8
	Race/Color	54.1	**49.6**	59.7	56.7	61.6
	Religion	74.8	**71.2**	76.6	73.9	81.1
	Sexual Orientation	86.0	**69.9**	88.2	86.0	**78.5**
	Socioeconomic Status	69.0	**59.5**	**69.5**	**69.5**	74.2
	Average	65.1	**56.0**	**65.6**	67.8	66.9