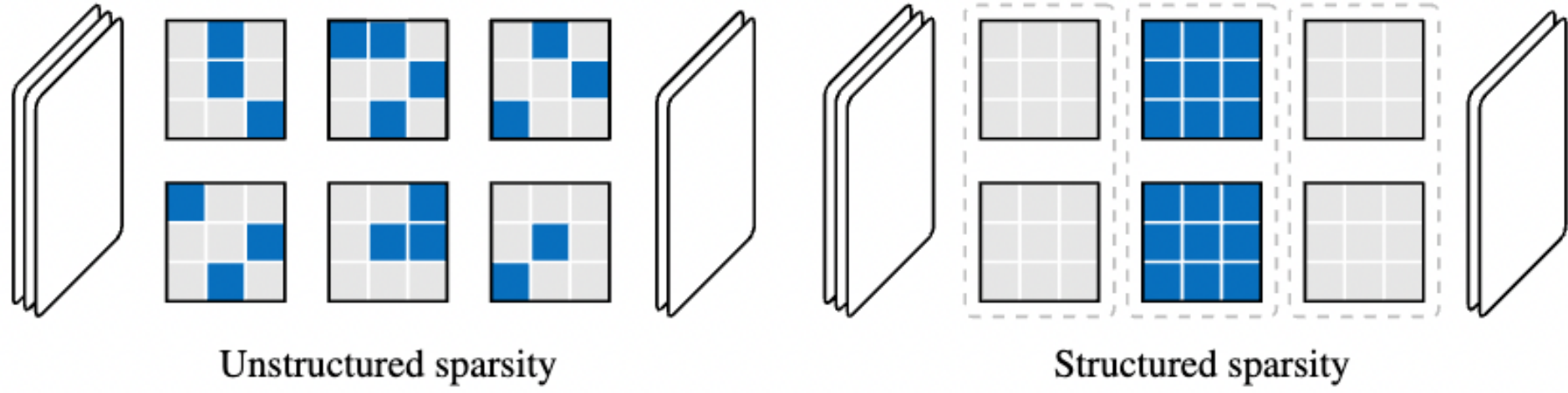


Empirical Upper Bounds for Unstructured Sparsity in Compute-Efficient Language Modeling

Esha Singh^{1,2}, Shane Bergsma², Nolan Dey², Joel Hestness², Gavia Gray²

UC San Diego, Department of Computer Science ¹, Cerebras Systems ²

Background: Unstructured Sparsity’s potential for compute-efficiency



If $\mathcal{W}$ contains all weights in a network, sparsity s :

$$s(\epsilon) = \frac{|\{w \in \mathcal{W} : w < \epsilon\}|}{|\mathcal{W}|} \quad \text{and} \quad \rho(\epsilon) = 1 - s(\epsilon)$$

Density ρ is proportion of weights greater than ϵ

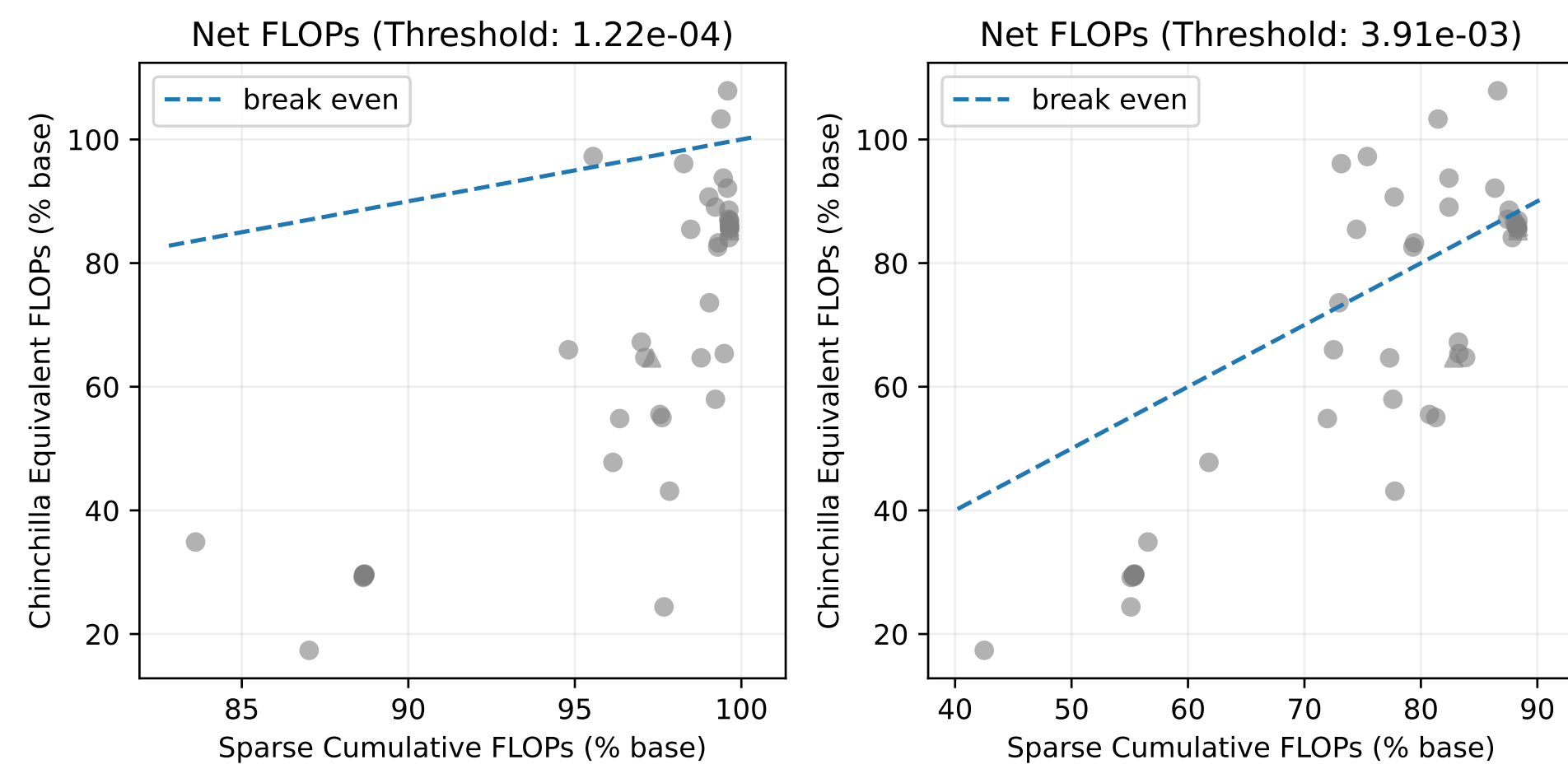
• Simplified Neural scaling law relating FLOP budget to validation loss,

$$\hat{L}(f) = \left(\frac{f}{f_b}\right)^{-\gamma} + E$$

f_b : base FLOPs, γ : exponent fitted to data

- Sparsity in DNNs reduces computational costs via FLOPs reductions both during training and inference.
- Current sparse NN studies typically examine trade-offs in final models rather than scaling laws.
- Goal: Empirical UB advantage of sparsity for pre-training LLMs.

We posit that an *upper-bounded* advantage is an effective way to rank various sparsification methods



Sparse Cumulative FLOPs for $\epsilon = 2^{-13}/2^{-8}$ (left/right) and compare that to the FLOPs required to train a compute optimal model to the same validation loss, defined as Chinchilla Equivalent FLOPs. For sparsity to be worthwhile it’s necessary to be above the “break even” line

1. Our work focuses on how regularization and parametrization affects sparsity during training, for example: L_1 regularization is known to induce a Laplace distribution in the weights.
2. Based on our existing scaling laws, we evaluate potential benefits of unstructured sparsity - where zeros in the weight matrix cost zero flops.

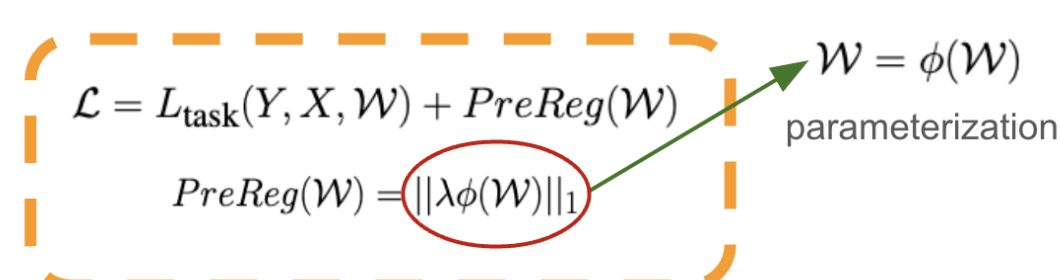
Research Question: Can we characterise an empirical upper bound on the advantage of sparsity for pre-training LLMs ?

Partial answers in existing work

- [Mishra et al., 2021, Cerebras Systems, 2024]: Development of hardware that can accelerate sparsity
- [Neural Magic, 2024]: software with improved existing hardware
- [Bambhaniya et al., 2024, Frantar and Alistarh, 2023]: new algorithms that increase sparsity during training or after.

Characterization via: Sparsifying Regularization

- *Parametrization*: We study methods whose interaction with the optimization is characterized by a parametrization and update rules. If weights $\mathcal{W}$, the parametrization, ϕ , then, $y = f(x, \Phi(\mathcal{W}; \mathcal{U}))$
- *Update rules*: refer to two specific functions: *PreReg* and *Update*



Example of sparsifying regularizers that we study. η is learning rate and λ is regularization weight

Empirical Upper Bounds for Sparse Scaling As threshold increases, our upper bound becomes less precise since it ignores performance impact of zeroing weights.

Sparse Pre-training Advantage: Inverting simplified neural scaling law, dictates that for any increase in loss, a smaller model could achieve equal results with fewer FLOPs,

$$\hat{f}(L) = \exp\left(\log(f_b) - \frac{\log(L - E)}{\gamma}\right)$$

Thus, for non-trivial pre-training, it needs to cost fewer FLOPs, than this, Then pre-training Advantage: $\Delta f = \hat{f}(L) - s(\epsilon, \bar{\rho})$

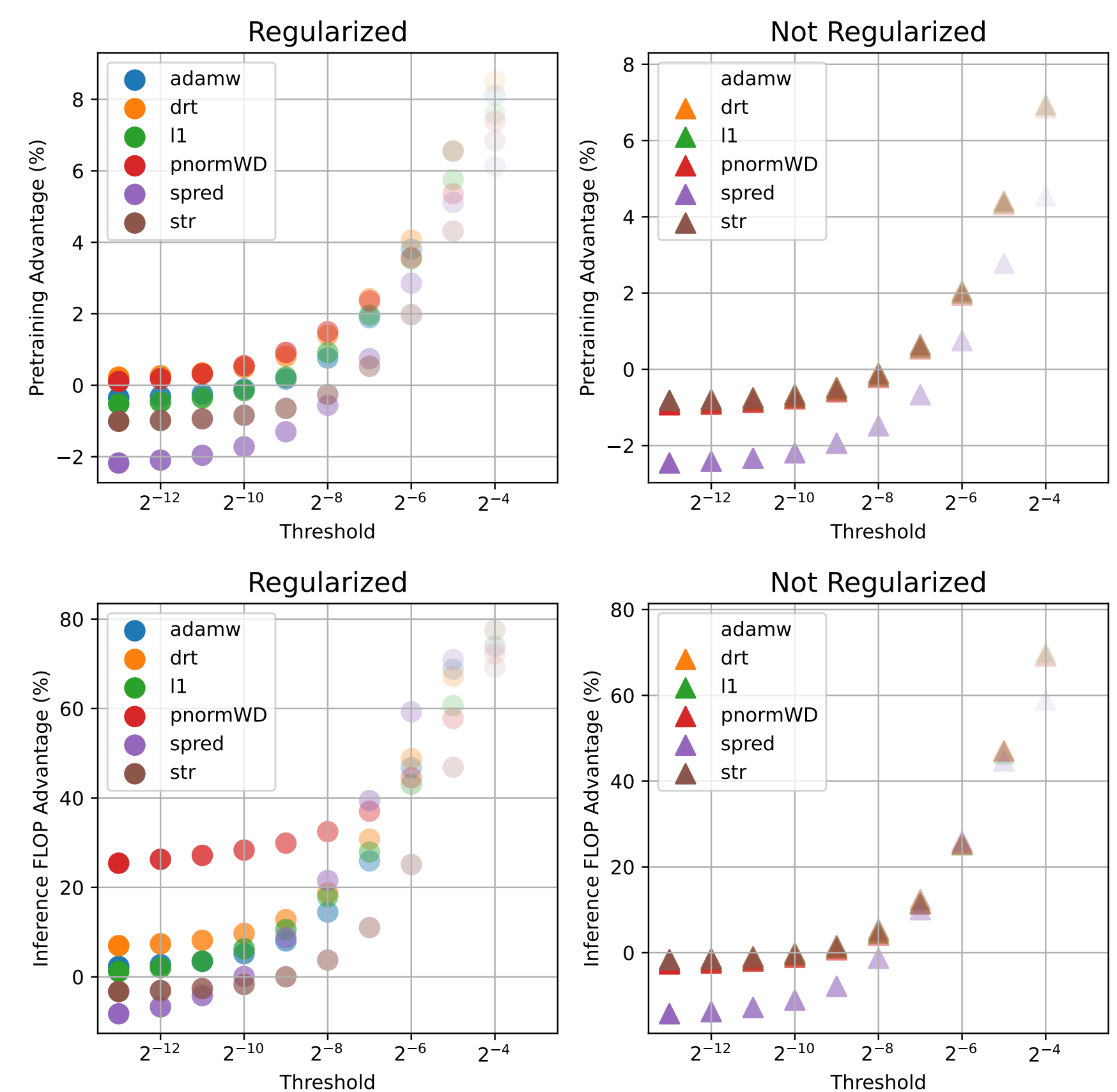
Sparse Inference Advantage: Similar relationship holds for sparsity in final pre-trained model. For given validation loss, to infer FLOPs per token $\hat{f}_d$, in we use the following scaling law,

$$\hat{f}_d = 2.6198 \times 10^{12} \times L^{-8.94243}$$

Then to compare final FLOPs per token using the density of final checkpoint ρ_D , i.e *Inference Advantage*,

$$\Delta \hat{f}_d = \hat{f}_d - s_d(\epsilon, \rho_D)$$

Experiments: We replicate Cerebras-GPT (111M parameters) trained on the Pile dataset for different sparse regularizers.



Upper bound performance of sparsifying regularizers, showing optimal hyperparameter configurations (left) and unregularized baseline models (right).

Conclusion We evaluated LLM sparsity using scaling laws, finding an optimal training configuration that offers pre-training FLOP savings and 30-40% inference speedup!

Bibliography

- [Bambhaniya et al., 2024] Bambhaniya, A. R., Yazdanbakhsh, A., Subramanian, S., Kao, S.-C., Agrawal, S., Evci, U., and Krishna, T. (2024). Progressive gradient flow for robust N:M sparsity training in transformers. *arXiv preprint arXiv:2402.04744*.
- [Cerebras Systems, 2024] Cerebras Systems (2024). Train a model with weight sparsity. Cerebras Systems Documentation. Version 2.1.1.
- [Frantar and Alistarh, 2023] Frantar, E. and Alistarh, D. (2023). SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337.
- [Mishra et al., 2021] Mishra, A., Latorre, J. A., Pool, J., Stosic, D., Stosic, D., Venkatesh, G., Yu, C., and Micikevicius, P. (2021). Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378*.
- [Neural Magic, 2024] Neural Magic (2024). DeepSparse. GitHub repository.