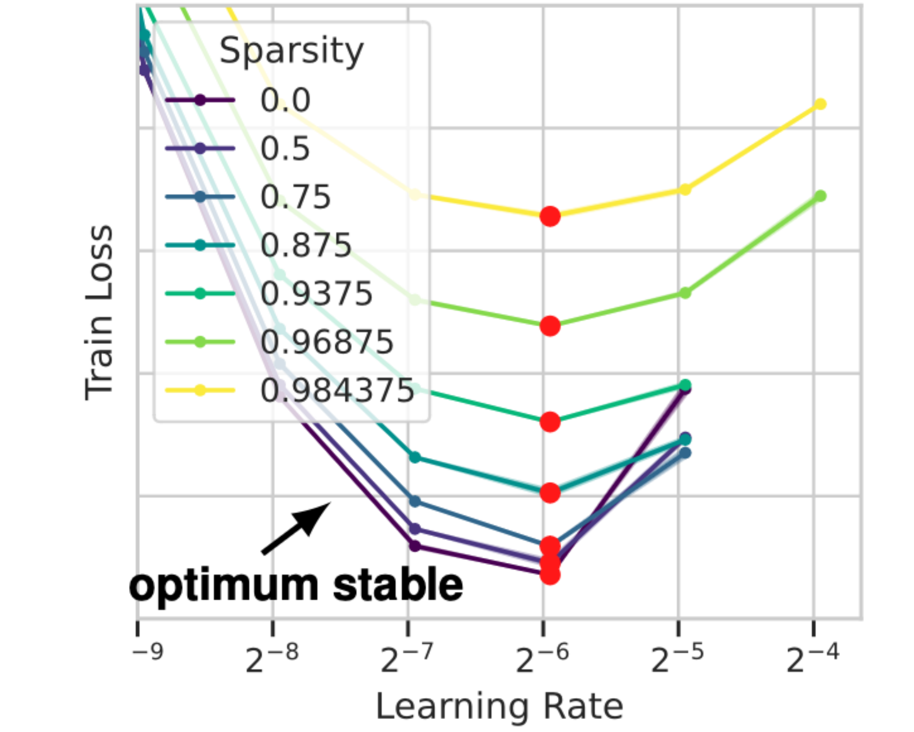
Sparse maximal update parameterization: A holistic approach to sparse training dynamics

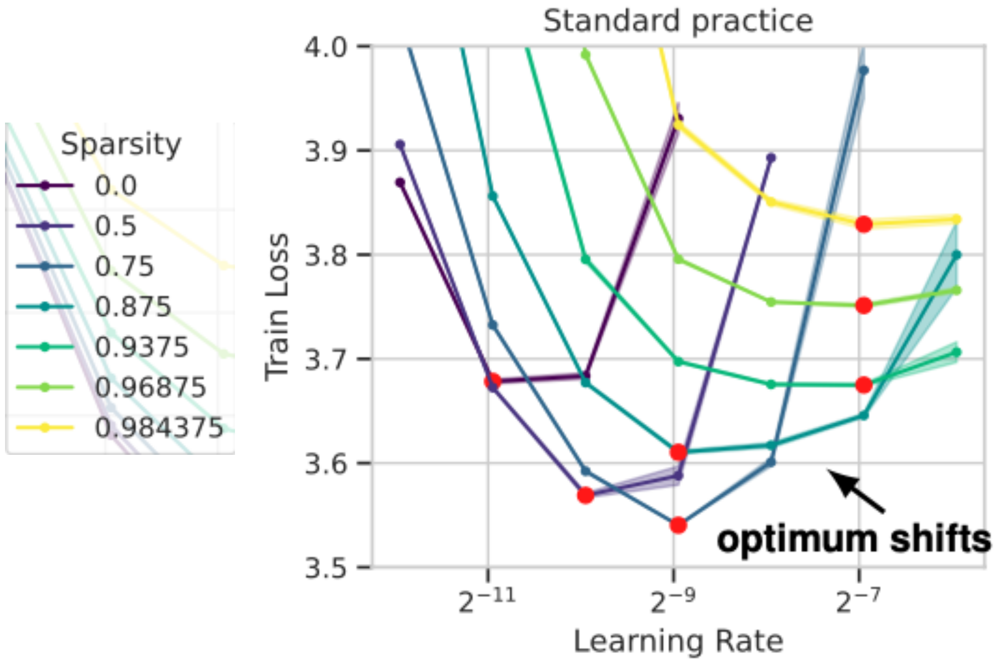
Nolan Dey, Shane Bergsma, Joel Hestness

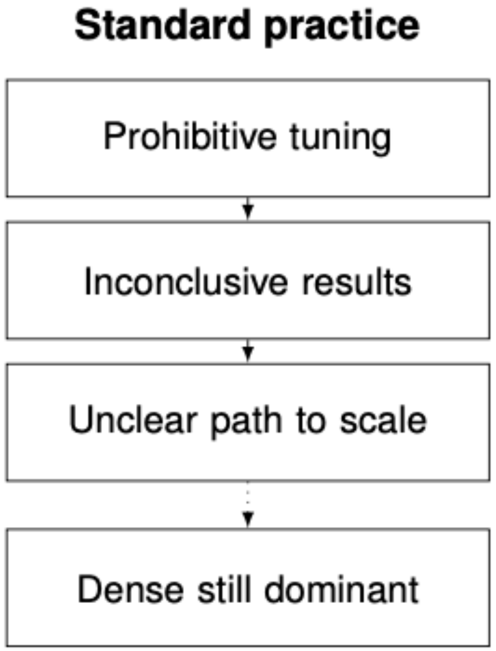
TL;DR: We introduce the sparse maximal update parameterization (SμPar) which ensures stable optimal HPs for any width or sparsity level. This dramatically reduces sparse HP tuning costs, allowing SμPar to achieve superior losses.



Motivation

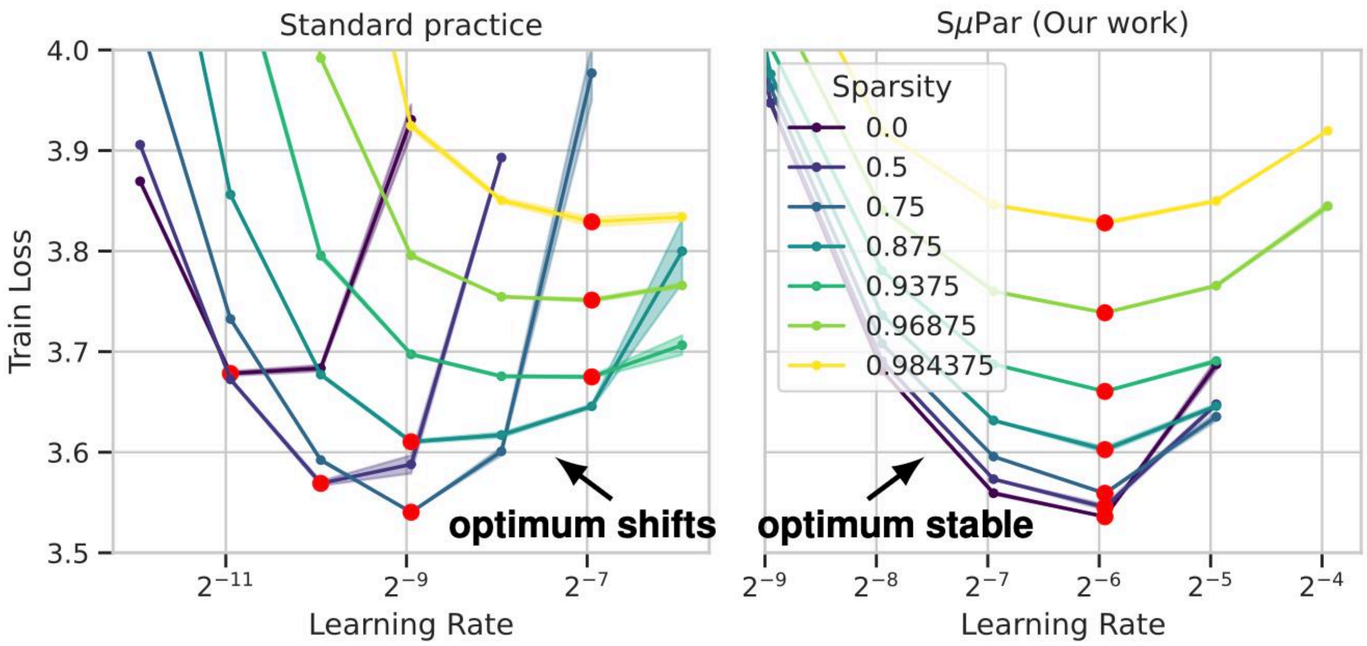
- When training sparse models, it is standard practice to **re-use the dense hyperparameters** (HPs)
- **Left:** Optimal HPs **systematically vary with sparsity level**
- Conducting a robust sparsity study would require retuning HPs for each sparsity level
- **Right:** Without stable optimal HPs across sparsity levels, it is **prohibitive** to robustly study large-scale sparse training

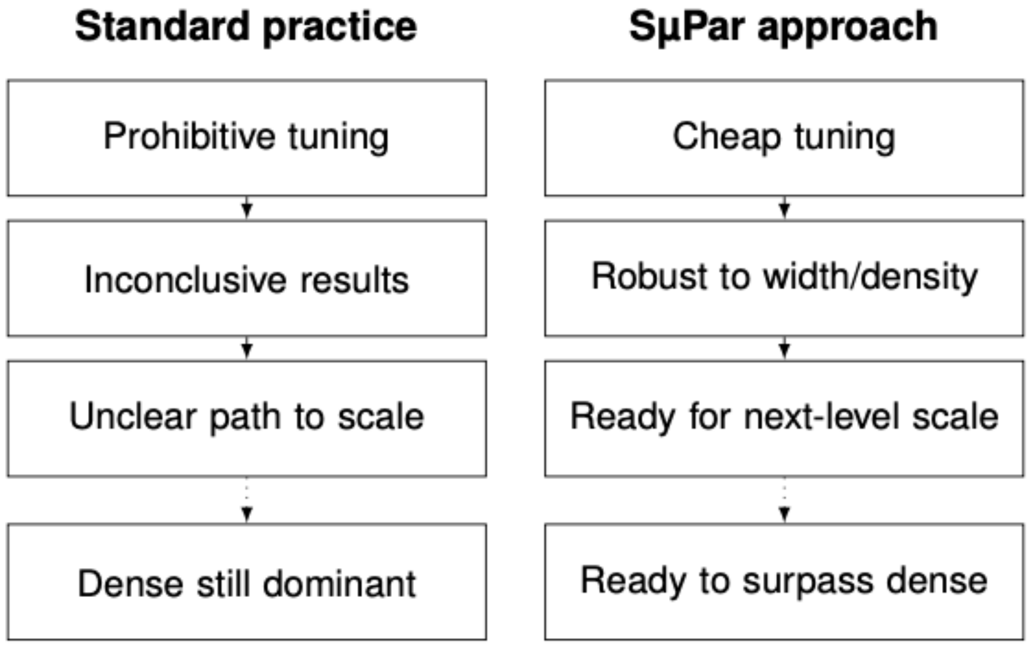




SuPar enables more robust sparse research

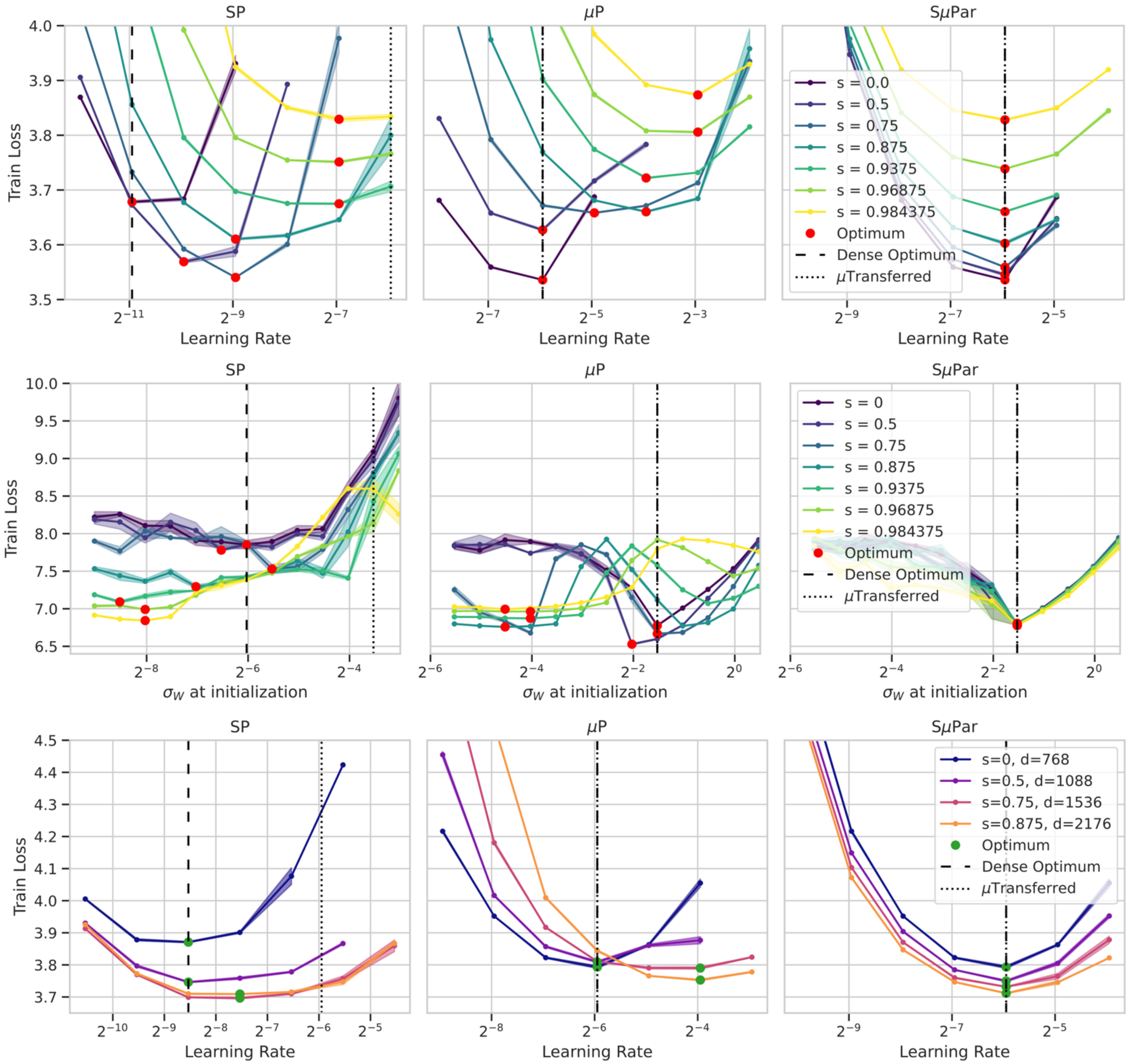
- **Left:** We propose Sparse Maximal Update Parameterization (**SμPar**), which enables the same HP values to be optimal as we vary **both** sparsity level and model width
- **Right:** SμPar enables more robust sparsity research
- In prior research that re-used dense HPs, sparse models are unfairly disadvantaged and these studies **merit re-examination**





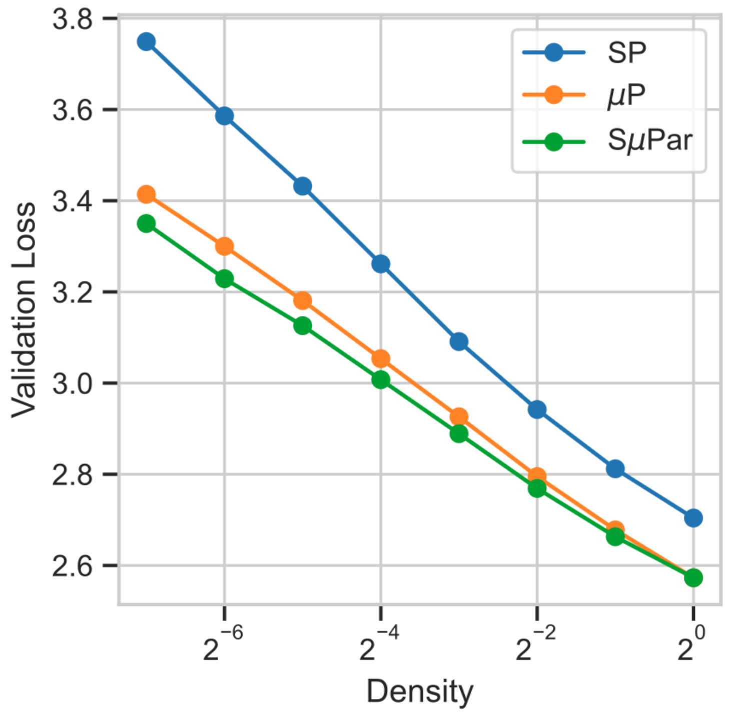
Static Sparsity Hyperparameter Transfer

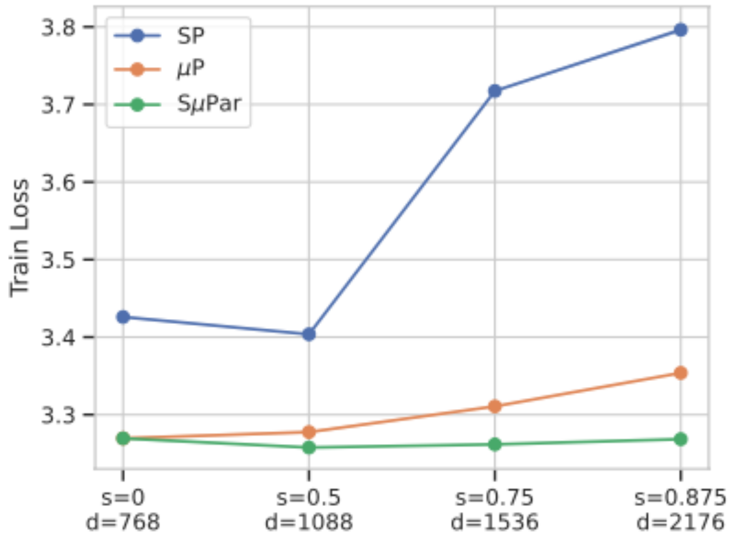
- Unlike SP and μP [1], SμPar enables **optimal HP transfer for any width or sparsity**
  - **Top:** SμPar enables stable η^* for any sparsity.
  - **Middle:** SμPar enables stable σ_W^* for any sparsity.
  - **Bottom:** SμPar enables stable η^* for any width and sparsity.
- Our **dense-tuned HPs perfectly transfer** to SμPar models ("μTransferred" vertical line)



Sparse LLM Pretraining

- Large networks trained with SμPar improve over SP and μP due to improved tuning
- **Top:** Apply static sparsity to 610M parameter LLM trained on 12.2B tokens. SμPar models improve over SP and μP due to improved tuning
- **Bottom:** Iso-Parameter wide-sparse scale 111M parameter LLM trained on 1B tokens. SuPar enables wide-sparse models to match dense loss at high sparsity levels, unlike SP and muP

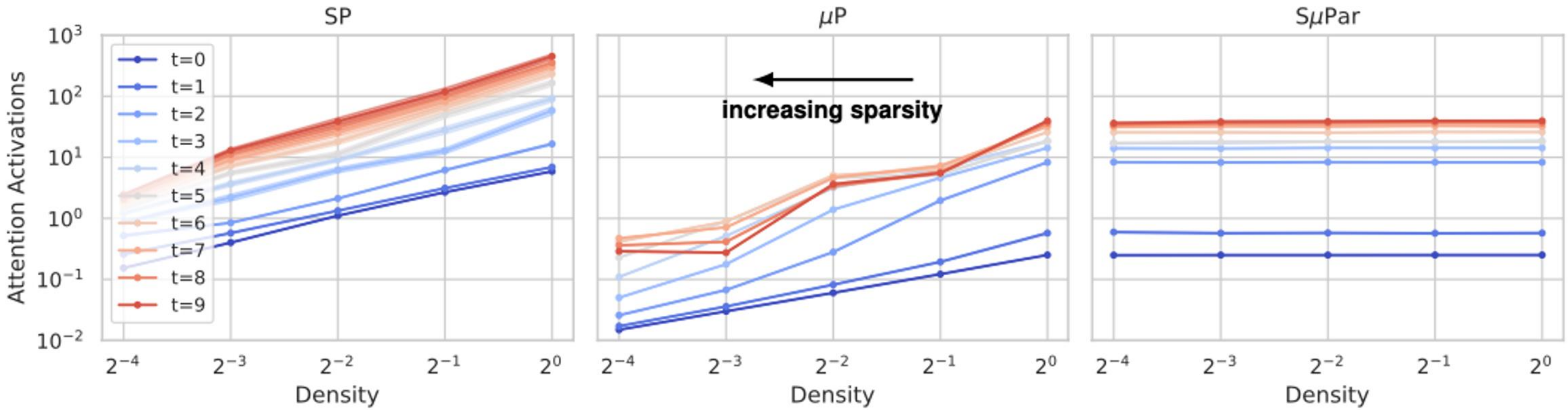




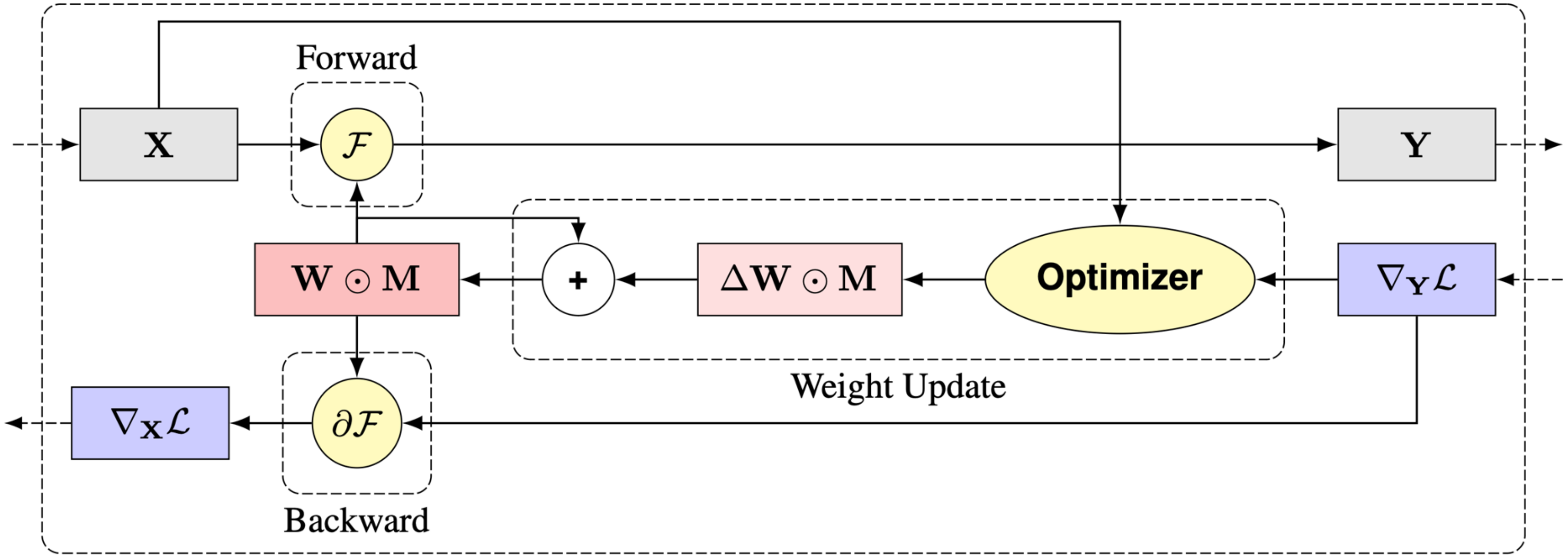
How SuPar works

SμPar stabilizes training dynamics

- **Setup:** For several sparsity levels, train a model for 10 steps and record activation L1 norm
  - All the points at each density value comprise a single training run
  - Each line has points from multiple models
- **Left & Middle:** For both SP and μP, sparsity causes vanishing activations and gradients
- **Right:** For SuPar, sparsity has little effect on activation scales and there is no vanishing.



Training step



If we apply sparsity to a linear layer (i.e., $\mathcal{F}$ is a fully-connected layer), our aim is to control:

1. **Forward pass:** $\mathbf{Y} = \mathcal{F}(\mathbf{X}, \mathbf{W} \odot \mathbf{M}) = \mathbf{X}(\mathbf{W} \odot \mathbf{M})$.
2. **Backward pass:** $\nabla_{\mathbf{X}}\mathcal{L} = (\nabla_{\mathbf{Y}}\mathcal{L}) \cdot (\mathbf{W} \odot \mathbf{M})^\top$.
3. **Effect of weight update $\Delta\mathbf{W}$ on $\mathbf{Y}$:** $\Delta\mathbf{Y} = \mathbf{X}(\Delta\mathbf{W} \odot \mathbf{M})$[1].

Sparse Maximal Update Parameterization (SμPar)

Feature Learning Desiderata (FLD): For layer l and token i, we desire that $\|\mathbf{Y}_i^l\|_2 = \Theta(\sqrt{d_{\text{out}}}), \|\nabla_{\mathbf{X}}\mathcal{L}_i^l\|_2 = \Theta(\sqrt{d_{\text{in}}}), \|\Delta\mathbf{Y}_i^l\|_2 = \Theta(\sqrt{d_{\text{out}}}), \forall i, \forall l.$

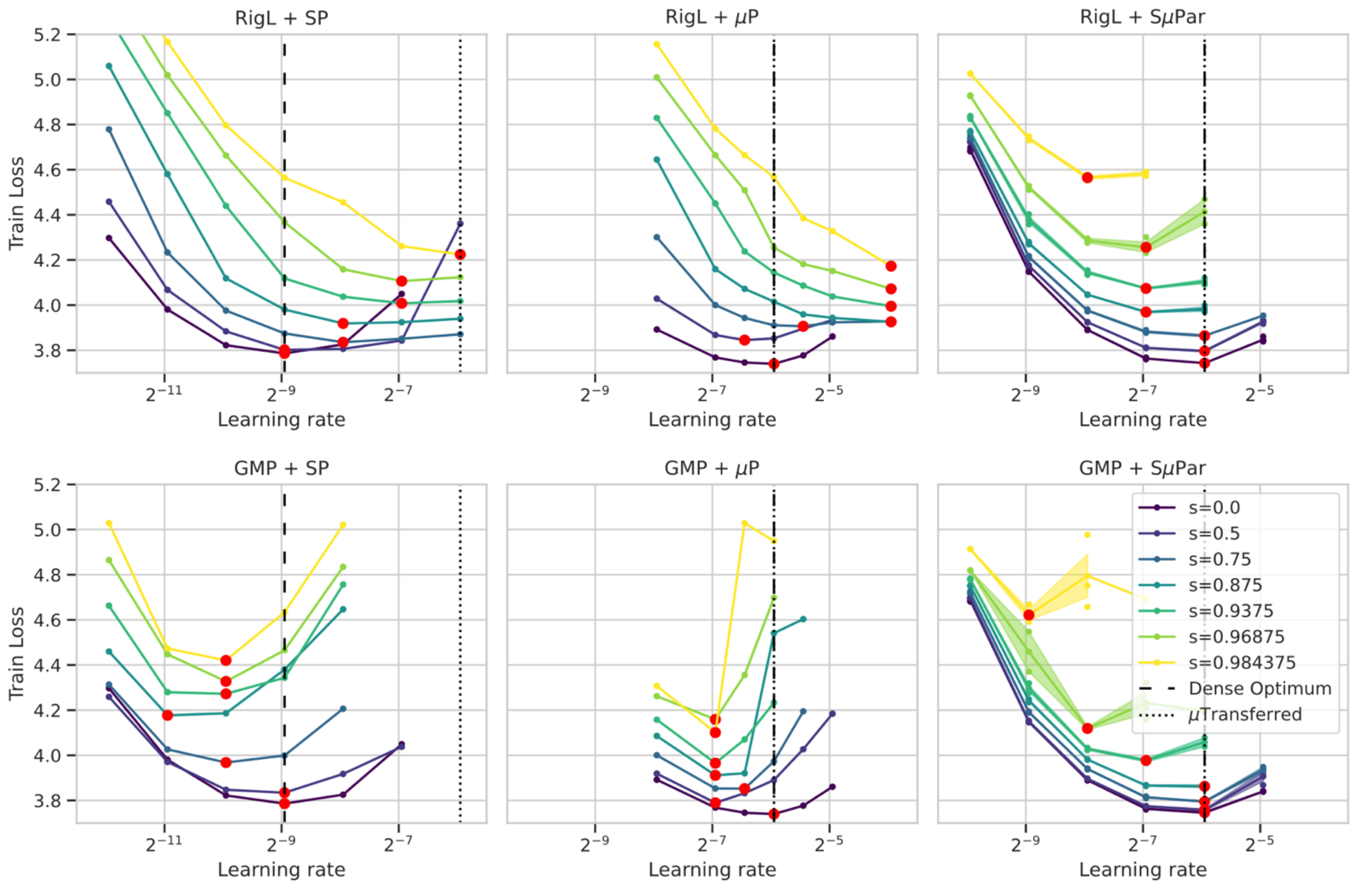
- SμPar ensures the typical element size of $Y, \nabla_X L, \Delta Y$ is $\Theta(1)$with respect to change in width m_d **and change in density m_ρ**, satisfying the FLD
- SμPar extends μP [1] for sparsity by applying corrections to hidden LR and initialization variances.
- Code: https://github.com/EleutherAI/nanoGPT-mup/tree/supar

Table 1: Summary of SP, μP, and SμPar

Parameterization	SP	μP	SμPar
Embedding Var.	σ^2_{base}	σ^2_{base}	σ^2_{base}
Embedding LR	η_{base}	η_{base}	η_{base}
Embedding Fwd.	$\mathbf{X}^0\mathbf{W}_{\text{emb}}$	$\alpha_{\text{input}} \cdot \mathbf{X}^0\mathbf{W}_{\text{emb}}$	$\alpha_{\text{input}} \cdot \mathbf{X}^0\mathbf{W}_{\text{emb}}$
Hidden Var.	σ^2_{base}	$\sigma^2_{\text{base}}/m_d$	$\sigma^2_{\text{base}}/(m_d m_\rho)$
Hidden LR (Adam)	η_{base}	η_{base}/m_d	$\eta_{\text{base}}/(m_d m_\rho)$
Unembedding Fwd.	$\mathbf{X}^L\mathbf{W}^\top_{\text{emb}}$	$\alpha_{\text{output}}\mathbf{X}^L\mathbf{W}^\top_{\text{emb}}/m_d$	$\alpha_{\text{output}}\mathbf{X}^L\mathbf{W}^\top_{\text{emb}}/m_d$
Attention logits	$\mathbf{Q}^\top\mathbf{K}/\sqrt{d_{\text{head}}}$	$\mathbf{Q}^\top\mathbf{K}/d_{\text{head}}$	$\mathbf{Q}^\top\mathbf{K}/d_{\text{head}}$

Dynamic sparsity hyperparameter transfer

- None of SP, μP, or SμPar achieve stable η^* across sparsity levels for RigL [2] (**Top**) or GMP [3] (**Bottom**)
- For SμPar, higher sparsity means lower η^* because SμPar is "overcorrecting".
- **Problem:** Dynamic sparse mask updates shift distribution of unmasked/non-zero weights to be non-Gaussian
- **Future work:** Generalize SμPar for dynamic sparsity



References

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

[2] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. (2020). **“Rigging the lottery: Making all tickets winners.”** In International conference on machine learning. PMLR, 2943–2952.

[3] Michael Zhu and Suyog Gupta. (2017). **“To prune, or not to prune: exploring the efficacy of pruning for model compression.”** arXiv preprint arXiv:1710.01878.