SD²: Self-Distilled Sparse Drafters

Mike Lasby*†, Nish Sinnadurai, Valavan Manohararajah,
Sean Lie, Yani Ioannou†, Vithursan Thangarasa

ICML
International Conference On Machine Learning

Motivation

- **Do sparse draft models improve speculative decoding?**
- Fine-grained sparsity is positioned between dense and layer-pruned draft models on the pareto front of accuracy and performance.
- Our prior work [1] demonstrated that self-data distillation effectively aligns layer-pruned draft models to a target model.
- Low latency draft models benefit speculative decoding; however, a high draft token acceptance rate must be maintained to be effective.
- Are dense, layer-pruned, or sparse draft models the best choice for maximizing acceleration with speculative decoding?

Key insights

- We introduce **SD²**, a novel methodology for obtaining fine-grained sparse draft models.
- We demonstrate the **superiority of fine-grained sparsity for accelerating speculative decoding** and downstream evaluation tasks compared with layer-pruned models.
- We showcase the **effectiveness of self-data distillation fine-tuning for model alignment**, even when aligning with a different model family with Universal Assisted Drafting (UAG).
- When paired with optimized sparse representations, we find that the end-to-end acceleration of speculative decoding with fine-grained sparse draft models is comparable to and **in some cases exceeds that of dense draft models, particularly with larger draft model sizes.**

Method

- **One-shot pruning:** We leverage SparseGPT [2] for one-shot pruning of the draft model.
- **Layer-pruning:** We prune sequential decoder blocks with the smallest angular distance between their inputs and outputs following [3].
- **Self-data distillation:** Generate datasets by prompting the target model with the concatenated inputs and outputs from SFT datasets. The target model outputs are extracted as labels.
- **Sparse fine-tuning:** The pruned draft models are fine-tuned using the self-data distilled datasets with a static sparse mask.

Experimental design and analysis

- We prune and fine-tune draft models from the Llama-3.2 and Qwen-2.5 model families.
- Draft models are evaluated on SpecBench and OpenLLM V1 benchmarks
- We measure the latency of our draft models using nm-vLLM [4] and calculate the improvement factor as:

$$\text{Improvement Factor} = \frac{\text{MAL}}{kc+1},$$

Where MAL is the mean number of accepted tokens per round, k is the number of draft tokens speculated per round, and c is the ratio of target/draft model latency or MACs.

Results

Speculative decoding improvement factor

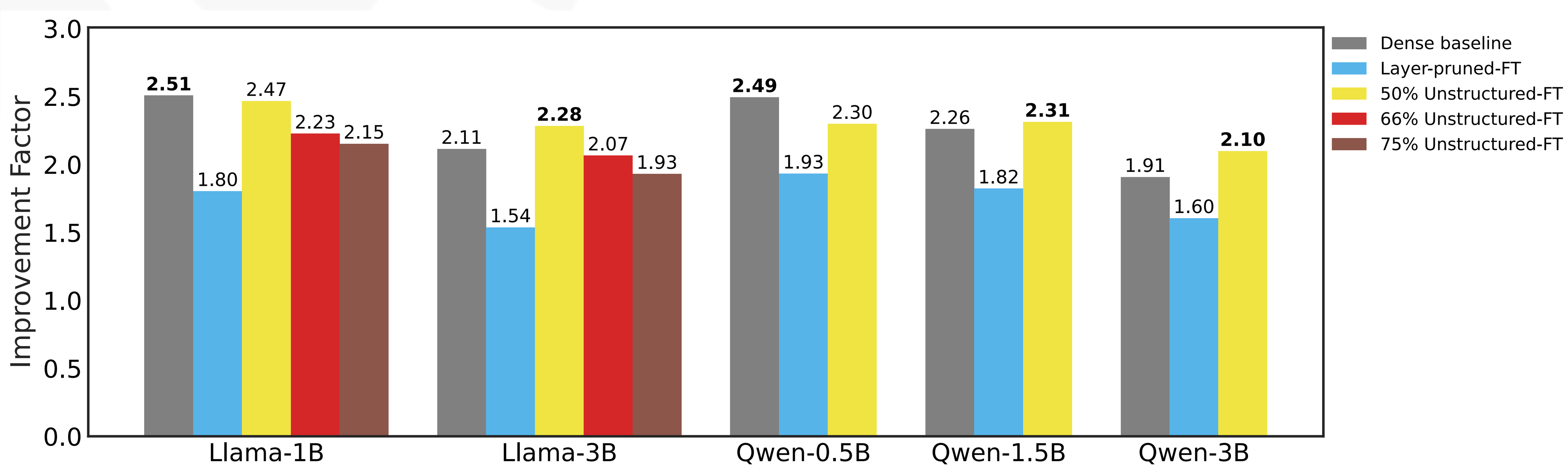

Figure 1: **Improvement factor of dense, layer-pruned, and SD² unstructured** Llama and Qwen models drafting for Llama-3.1-70B-Instruct and Qwen-2.5-72B-Instruct, respectively. SD² drafters outperform layer-pruned draft models and dense drafters in the 1.5B and 3B model size categories.

Universal assisted drafting

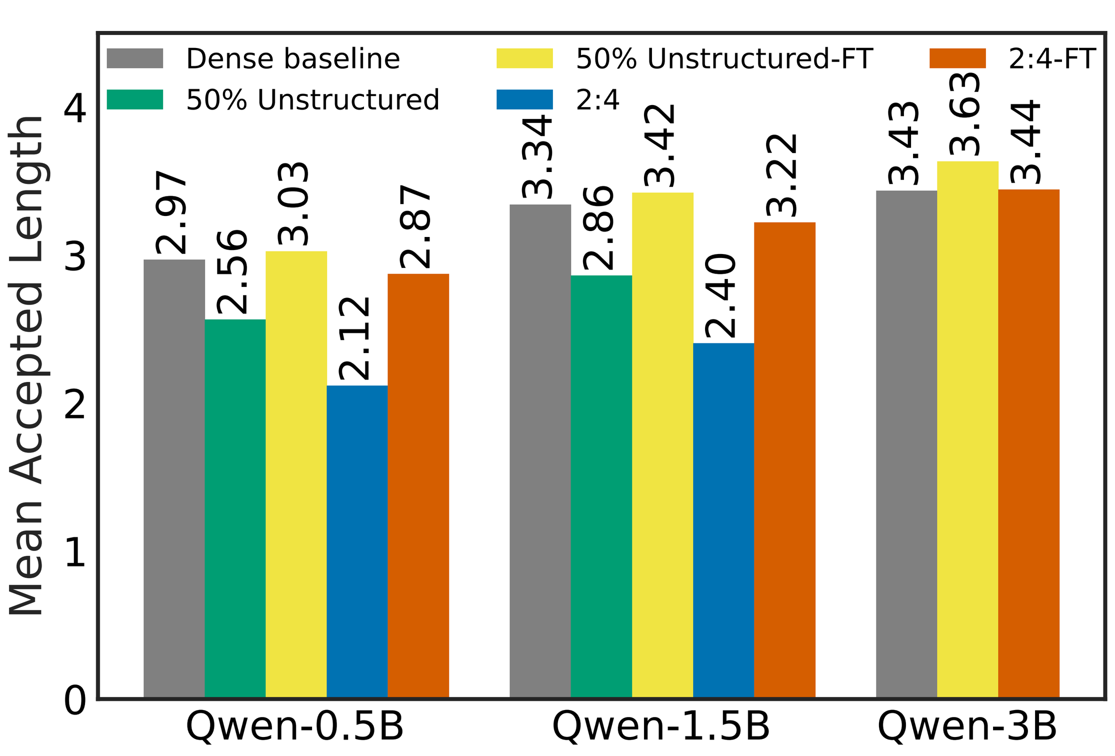

Figure 2: **SpecBench MAL for SD² Qwen-2.5 models drafting for Llama-3.1-70B-Instruct in the UAG setting.** These results illustrate the benefits of SD² for aligning draft models *even across different model families*. SD² Qwen drafters achieve a higher MAL than their dense counterparts.

MACs analysis

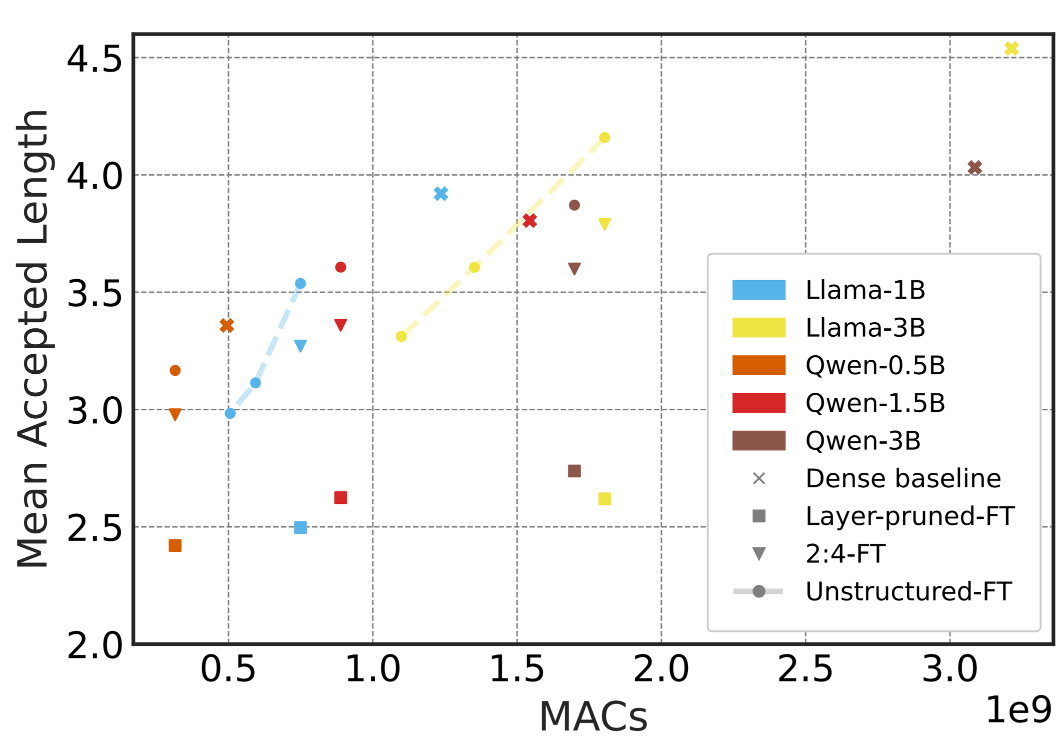

Figure 3: **MAL vs MACs for layer-pruned, dense, and sparse draft models.** Particularly notable are the Qwen-2.5 unstructured sparse drafters which approach iso-MAC performance compared to the dense models.

Pseudocode

Algorithm 1 SD^2: Self-distilled sparse drafters

Input: Draft model M_d with parameters θ, target model M_t, calibration dataset D_{cal}, supervised fine-tuning dataset D_{sft}, self-data distillation (SDD) context C, optimizer $\mathcal{O}$, learning rate α, number of iterations T, and batch size N.
Output: Fine-tuned sparse draft model, M'_d
Define $\textsc{SparsityHook}(\nabla_\theta \mathcal{L}_t, \theta, \theta_p)$
  for $p_i, \frac{\partial \mathcal{L}_t}{\partial p_i} \in \{\theta, \nabla_\theta \mathcal{L}_t\}$ **do**
    if $p_i \in \theta_p$ **then**
      $\frac{\partial \mathcal{L}_t}{\partial p_i} \leftarrow 0$
  return $\nabla_{\theta_s} \mathcal{L}_t$
end Define
$M'_d \leftarrow \text{SparseGPT}(M_d, D_{cal})$
$\theta_p \leftarrow \{p_i \in \theta \mid p_i = 0\}$
$D_{sdd} \leftarrow \emptyset$
for $\mathbf{X}_i, \mathbf{Y}_i \in D_{sft}$ **do**
  $\tilde{\mathbf{X}}_i \leftarrow \mathbf{C}||\mathbf{X}_i||\mathbf{Y}_i$
  $\tilde{\mathbf{Y}}_i \leftarrow M_t(\tilde{\mathbf{X}}_i)$
  D_{sdd}`.append(`$(\mathbf{X}_i, \tilde{\mathbf{Y}}_i)$`)`
M_d`.register(partial(SparsityHook(`θ, θ_p`)))`
for $t = 1$ **to** T **do**
  $\{(\mathbf{X}_n, \tilde{\mathbf{Y}}_n)\}_{n=1}^N \sim D_{self}$
  $C_t \leftarrow \sum_{n=1}^N \texttt{len}(\tilde{\mathbf{Y}}_n)$
  $\mathcal{L}_t \leftarrow \sum_{n=1}^N \sum_{j=1}^{\texttt{len}(\tilde{\mathbf{Y}}_n)} -\log M_d(\tilde{y}_{n,j} | \mathbf{X}_n, \tilde{\mathbf{Y}}_{n,:j-1})$
  $\mathcal{L}_t \leftarrow \mathcal{L}_t / C_t$
  ▷ `.backwards()` Triggers SparsityHook
  $\nabla_\theta \mathcal{L}_t \leftarrow \mathcal{L}_t$`.backwards()`
  $\theta \leftarrow \mathcal{O}(\theta, \nabla_\theta \mathcal{L}_t, \alpha)$
Return: M'_d

References

[1] V. Thangarasa, G. Venkatesh, M. Lasby, N. Sinnadurai, and S. Lie, "Self-Data Distillation for Recovering Quality in Pruned Large Language Models," presented at the Eighth Conference on Machine Learning and Systems, Feb. 2025.
[2] E. Frantar and D. Alistarh, "SparseGPT: Massive Language Models Can be Accurately Pruned in One-Shot," in *Proceedings of the 40th International Conference on Machine Learning*, PMLR, Jul. 2023, pp. 10323–10337.
[3] A. Gromov, K. Tirumala, H. Shapourian, P. Glorioso, and D. Roberts, "The Unreasonable Ineffectiveness of the Deeper Layers," presented at the The Thirteenth International Conference on Learning Representations, Oct. 2024.
[4] Neural Magic. 2024. Neural magic nm-vLLM inference engine.

* *Work completed while on internship Cerebras*
† *University of Calgary*