ICP: Immediate Compensation Pruning for Mid-to-high Sparsity

Supplementary Material

1. Theoretical Foundations

Large models are typically hierarchical and can be effectively viewed as a complex nested function:

$$F(x_i;\theta) = f_n\left(\dots f_j\left(\dots f_1\left(x;\theta_1\right)\dots;\theta_j\right)\dots;\theta_n\right), \quad (1)$$

where x is the input sample, f_j represents the mapping function of Block B^j and θ denotes the function parameters. Pruning a block effectively introduces a perturbation $\Delta\theta$ to that block's parameters. Taking B^j as an example, when pruning is applied such that $\theta_j = \theta_j + \Delta\theta$, the error introduced by this perturbation propagates from the output of B^j through to the final Block, leading to a deviation in the global model output:

$$\Delta\mathcal{Y}_j \approx \frac{\partial F}{\partial f_j}\cdot\frac{\partial f_j}{\partial \theta_j}\cdot\Delta\theta_j = \left(\prod_{i=j+1}^{n}\frac{\partial f_i}{\partial z^{i-1}}\right)\cdot\frac{\partial z^j}{\partial\theta_j}\cdot\Delta\theta_j, \quad (2)$$

where ∂z^j represents the output of B^j, in the model, and $\Delta\mathcal{Y}_j$ denotes the change in the model output caused by pruning B^j. From this equation, we observe that the closer the pruned block is to the model's output (*i.e.*, the further back in the model), the more directly the pruning affects the output. Conversely, the further the pruned block is from the output, the more difficult it becomes to control the resulting error. However, we can select a subsequent Block B^k after B^j to compensate for the changes introduced by pruning. If we adjust the parameters of B^k, its effect on the model output is similarly given by:

$$\Delta\mathcal{Y}_k = \left(\prod_{i=k+1}^{n}\frac{\partial f_k}{\partial z_{k-1}}\right)\cdot\frac{\partial z^k}{\partial\theta_k}\cdot\Delta\theta_k. \quad (3)$$

By setting $\Delta\mathcal{Y}_j + \Delta\mathcal{Y}_k = 0$, we can derive:

$$\Delta\theta_k = -\frac{\left(\prod_{i=j+1}^{n}\frac{\partial f_i}{\partial z^{i-1}}\right)\cdot\frac{\partial f_j}{\partial\theta_j}\cdot\Delta\theta_j}{\left(\prod_{i=k+1}^{n}\frac{\partial f_i}{\partial z^{i-1}}\right)\cdot\frac{\partial f_k}{\partial\theta_k}}. \quad (4)$$

From Eq. (4), we can observe that the closer B^k is to B^j, the more direct the relationship between their parameters. Specifically, if $k = j+1$, then $\prod_{i=j+1}^{k}\frac{\partial f_i}{\partial z^{i-1}} = \frac{\partial f_{j+1}}{\partial z^j}$, making the compensation more direct and easier to control.

However, the above compensation method requires access to the model's output, which can be both memory-intensive and time-consuming. Therefore, Progressive Block Pruning adopts a partial compensation approach. During error compensation, we directly use the MSE loss of z^k, the output of B^k, as the compensation loss. This can be expressed as:

$$\mathcal{L}_k(\theta_k) = \frac{1}{2m}\sum_{i=1}^{m}\|z'^k(\theta'_k) - z^k(\theta_k)\|^2. \quad (5)$$

The global optimization objective for the model is:

$$\mathcal{L}_{global} = \frac{1}{2m}\sum_{i=1}^{m}\|F'(x_i) - F(x_i)\|^2. \quad (6)$$

The optimization directions of Eq. (5) and Eq. (6) are aligned because, when we take the derivatives of these two equations, we obtain:

$$\frac{\partial\mathcal{L}_k(\theta_k)}{\partial\Delta\theta_k} = \frac{1}{m}\sum_{i=1}^{m}\left(\Delta z'^k(x_i)\cdot\frac{\partial z^k}{\partial\theta_k}\right), \quad (7)$$

$$\frac{\partial\mathcal{L}_{\text{global}}}{\partial\Delta\theta_k} = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{\partial F}{\partial z^k}\cdot\Delta z'^k(x_i)\cdot\frac{\partial z^k}{\partial\theta_k}\cdot\frac{\partial F}{\partial z^k}\right). \quad (8)$$

The signs of these derivatives are the same, indicating that the optimization directions are consistent. Therefore, we can use $\mathcal{L}_k$ to implicitly optimize $\mathcal{L}_{global}$.

Based on the above theoretical analysis, we can draw the following three conclusions:

- The error introduced by pruning can be compensated for by adjusting subsequent blocks in the model.
- The closer the compensating block is to the pruned block, the more direct the relationship between their parameters.
- Local MSE can be used to implicitly optimize the global MSE.

2. Implementation Details

This subsection supplements the implementation details of pruning and compensation processes of ICP based on experimental setup (Section 4.1) outlined in the main text.

2.1. Algorithm Implementation

ICP adopts an iterative block-wise pruning-compensation strategy. During pruning, a one-shot approach is applied-each block is pruned directly to the target sparsity level in a single step rather than progressively over multiple stages[1, 3], considering computational efficiency and overfitting prevention. In the compensation phase, weight updates are restricted to the currently processed block; no other blocks, including the recently pruned one, participate in weight updates. Moreover, a unified learning rate is employed for all compensation processes across blocks, obviating the need for block-specific learning rate tuning.

2.2. Memory Efficiency

ICP is designed to be memory-efficient. Since both pruning and compensation operations are performed on different single block, GPU memory only needs to store the weights of a single block at any given time. Additionally, the optimizer state and gradients are similarly constrained to a single block. This significantly reduces memory usage despite involving training-like updates.

At the outset, the input z of the first block is copied to produce z' beyond this point, no additional storage space is allocated for any intermediate results at any stage, including during compensation. As the sliding window advances, z and z' are updated in place rather than storing intermediate forward pass results.

During compensation, when z' through a block to generate labels for z, all samples in z' are processed in a single forward pass an updated. This avoids the per-iteration forward pass required in traditional knowledge distillation, which needs to maintain two models simultaneously. Consequently, only one block's parameters are stored in GPU memory.

Finally, all intermediate results are offloaded to system memory. GPU inference of the current batch and preloading of the next batch are executed in parallel to further optimize runtime.

3. Supplementary Results of Unstructured Pruning

In Section 4.2 of the main text, we reported the instance segmentation performance of the pruned SAM model using the proposed method under single-point interaction. Here, we supplement the segmentation performance of the SAM model using bounding box interaction in Tables 1 and 2. The experimental setup strictly adheres to the configurations described in the main text, and the implementation details are fully consistent with those outlined in the previous section. Compared to single-point interaction, bounding box interaction further narrows the performance gap between different compression methods. However, the advantage of our method over the baseline remains significant.

4. Additional Ablation Studies

4.1. Calibration Dataset Sensitivity Ablation

A key step in the proposed ICP framework involves randomly sampling 128 instances from the pre-training dataset to construct a calibration set. In Section 4.5, we investigated the impact of dataset size on the model's performance. Here, we extend our analysis by conducting ablation studies on different calibration sets-each consisting of 128 samples randomly drawn from the same pre-training dataset-to assess how variations in the calibration data affect the model. We conducted 30 repetitions of the pruning process for Wanda, SparseGPT, and ICP on OPT-125M on spartsity of 50%, each time re-sampling the calibration set and recording the model's performance. As shown in Fig.1, different calibration sets induce minor fluctuations in the performance of the pruned model across the methods, with the ICP approach exhibiting slightly larger variance than Wanda and SparseGPT. This is attributed to the small size of the calibration set and the fact that the same calibration set is employed for both pruning and compensation. Consequently, certain calibration sets can deviate significantly from the overall pre-training dataset, and such deviations are further amplified during the compensation process—especially on the PTB dataset, which is not used for calibration. Nonetheless, the overall perplexity of ICP remains lower than that of Wanda and SparseGPT.

Table 1. Instance segmentation performance of SAM models (IoU, %) at various sparsity using box interaction on SA-1B. The models were calibrated on a subset of 128 images sampled from SA-1B.

Method	Model	10%	20%	30%	40%	50%	60%	70%	80%	90%
Magnitude	SAM-H	92.47	92.40	92.18	91.73	90.64	87.66	76.49	66.76	63.23
SparseGPT		92.48	92.47	92.44	92.35	92.12	91.50	89.77	85.02	76.10
Wanda		92.48	92.47	92.41	92.21	91.72	90.20	85.36	78.49	72.78
ICP (ours)		**92.48**	**92.48**	**92.46**	**92.38**	**92.29**	**91.92**	**91.52**	**90.18**	**84.67**
Magnitude	SAM-L	**92.07**	92.00	91.79	91.25	90.09	87.27	81.23	72.26	65.88
SparseGPT		92.06	**92.04**	91.99	91.87	91.57	90.87	88.90	83.93	75.33
Wanda		92.06	**92.04**	91.96	91.73	91.15	89.47	84.95	77.93	69.20
ICP (ours)		**92.07**	**92.04**	**92.00**	**91.93**	**91.78**	**91.50**	**90.86**	**89.14**	**83.50**
Magnitude	SAM-B	**89.83**	89.75	89.45	88.69	87.37	84.81	79.41	72.47	65.16
SparseGPT		**89.83**	**89.81**	89.73	89.57	89.17	88.36	86.54	82.37	76.61
Wanda		**89.83**	89.80	89.67	89.26	88.38	86.56	82.37	76.52	71.44
ICP (ours)		**89.83**	**89.81**	**89.76**	**89.70**	**89.47**	**89.12**	**88.39**	**86.62**	**82.23**

Table 2. Instance segmentation performance of SAM models (IoU, %) at various sparsity using box interaction on COCO. The models were calibrated on a subset of 128 images sampled from SA-1B.

Method	Model	10%	20%	30%	40%	50%	60%	70%	80%	90%
Magnitude	SAM-H	77.24	77.21	77.14	77.03	76.62	74.62	65.23	55.06	54.78
SparseGPT		**77.25**	77.24	77.21	77.16	77.07	76.78	75.99	72.59	66.40
Wanda		**77.25**	77.24	**77.23**	77.21	76.98	75.89	71.89	66.61	60.96
ICP (ours)		**77.25**	**77.25**	77.22	**77.22**	**77.02**	**76.87**	**76.38**	**75.08**	**70.82**
Magnitude	SAM-L	77.08	77.07	76.97	76.74	76.19	74.33	69.60	61.60	57.08
SparseGPT		77.09	77.07	**77.06**	**77.04**	76.89	76.54	75.44	72.31	65.72
Wanda		**77.10**	**77.08**	77.02	76.91	76.64	75.59	72.03	66.48	59.03
ICP (ours)		77.05	77.05	**77.06**	**77.04**	**76.91**	**76.63**	**76.08**	**74.55**	**70.62**
Magnitude	SAM-B	75.59	**75.59**	75.57	75.34	74.61	72.95	68.48	61.74	53.54
SparseGPT		**75.60**	75.57	75.55	75.51	75.34	74.85	73.87	70.82	66.59
Wanda		75.58	**75.59**	75.58	75.38	74.76	73.35	70.10	65.90	60.66
ICP (ours)		75.57	75.57	**75.59**	**75.54**	**75.39**	**75.03**	**74.69**	**73.64**	**70.55**

5. Inference Speed Acceleration Evaluation

Regarding the acceleration performance of the pruned models, we provide the following discussion. Although Wanda [4], SparseGPT [2], and ICP adopt different pruning strategies, the resulting models share the same sparsity pattern, yielding either unstructured or semi-structured sparse models. Consequently, during inference, the accel-

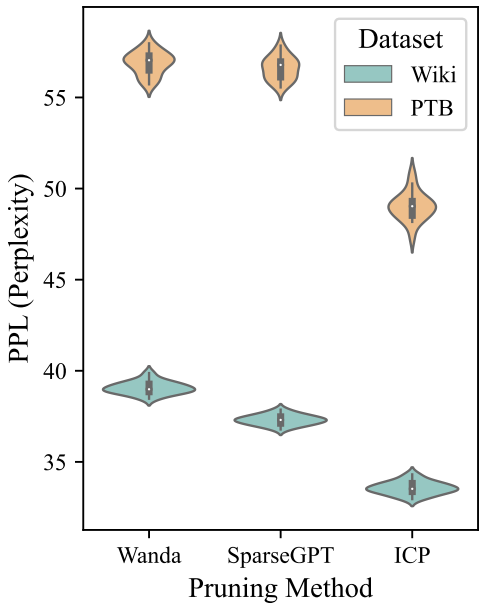

Figure 1. Fluctuations in model performance using different calibration datasets

eration effects achieved by the proposed ICP method are consistent with those of Wanda and SparseGPT. Experimental results corroborate that the acceleration performance of ICP is in line with our expectations. For reference, we cite the acceleration performance reported for SparseGPT: For OPT-125M, 2:4 sparse will get $1.64\times$ speedup on A100, $40/50/60/70\%$ sparsity, will get $1.57/1.82/2.16/2.39\times$ speedup on i9-7980XE.

References

[1] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018. 1

[2] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR, 2023. 2

[3] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015. 1

[4] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. 2