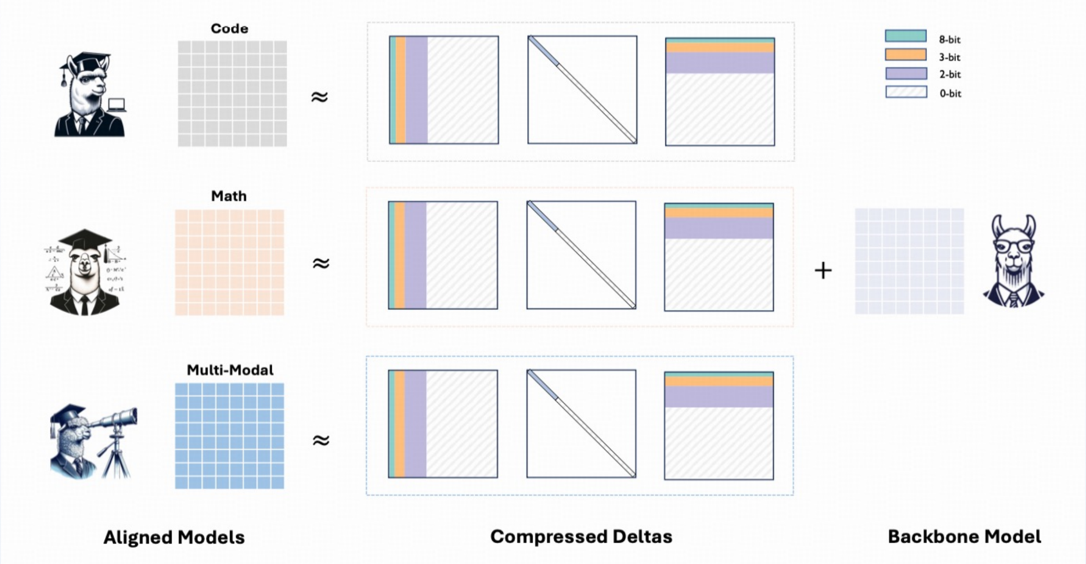
Delta-CoMe: Training-Free Delta-Compression with Mixed-Precision for Large Language Models

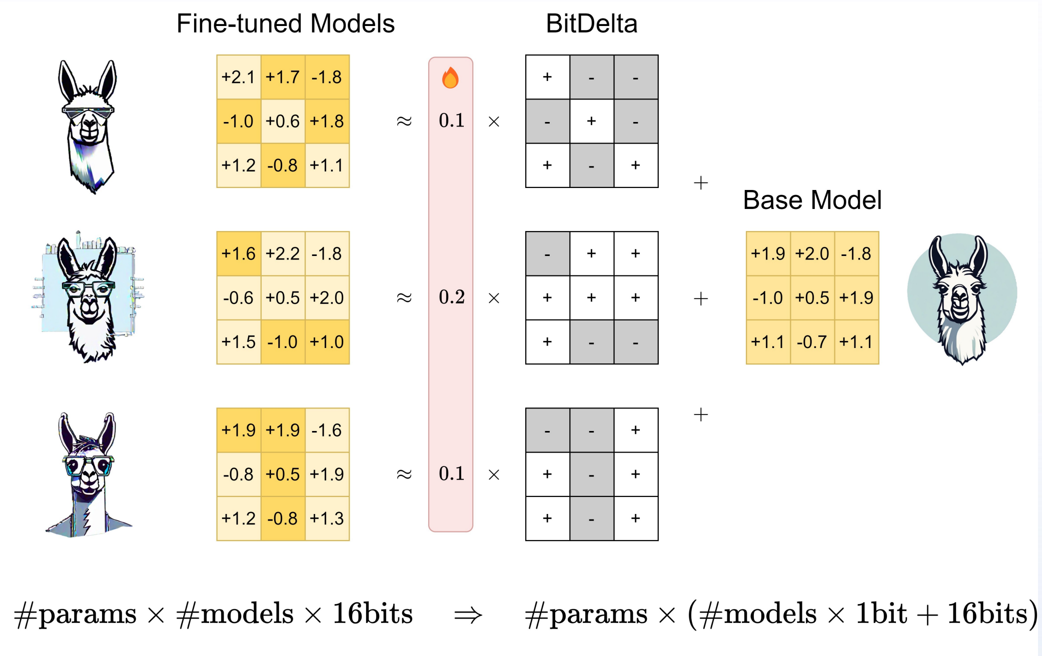


Delta-CoMe

Background: Delta obtained through fine-tune model and base model (et. Finetuned - base) can be compressed.

Related Work: Most recently, Bitdelta[1] compress Delta into 1-bit， and perform post-training to restore performance in several text benchmarks.

Existing Issues: Bitdelta failed in math, code which needs alignment that is of great significance and needs great efforts.



Method

- Delta-CoMe combine low-rank and low-bit
- Empirically, employing low-rank Delta can still retain down-stream performance
- Observing long-tail distribution after low-rank, Delta-CoMe mix-precision compression, assigning high bits to for singular vectors corresponding to larger singular values.

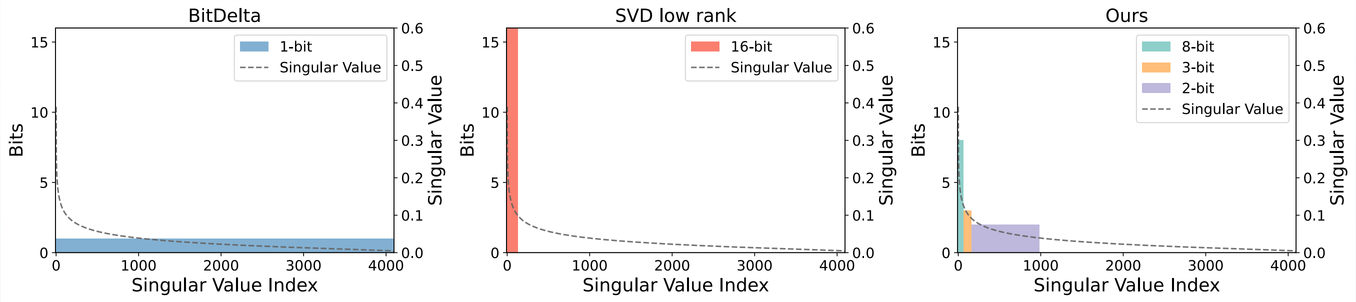


Figure 1: **Left**: illustration of BitDelta (Liu et al., 2024b), which employs 1-bit quantization for all the delta weights. **Middle**: illustration of low-rank compression (Ryu et al., 2023b), retaining the top-k singular values and the corresponding singular vectors. **Right**: illustration of the proposed Delta-CoMe method, which represents the singular vectors of larger singular values using high-bit vectors while compressing the singular vectors of smaller singular values into low-bit representations. This method is inspired by the long-tail distribution of singular values in delta weights.

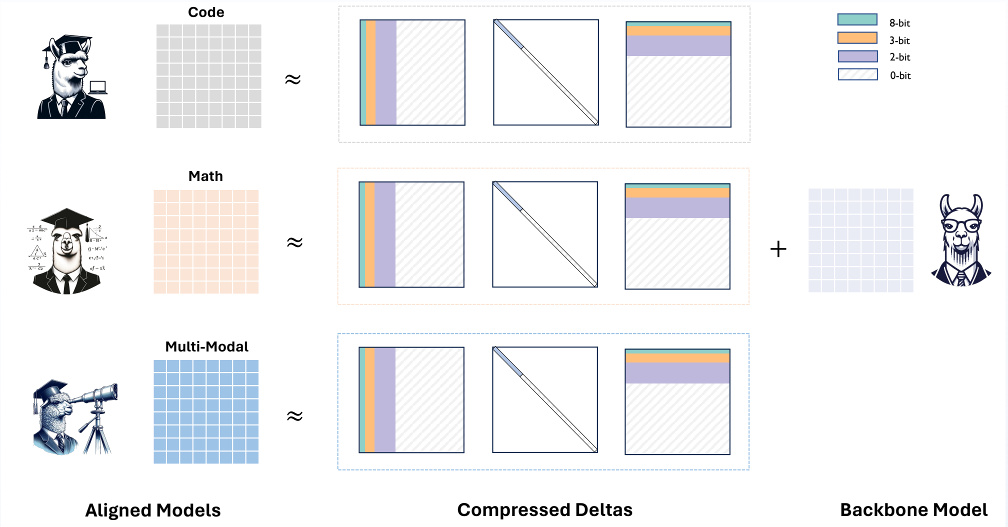


Figure 2: Illustration of Delta-CoMe, where we utilize varying bit-widths for singular vectors with different singular values. Singular vectors corresponding to larger singular values are assigned higher bit-widths. For extremely small singular values, we omit the singular vectors (i.e., 0-bit).

Method

Empirically, delta has low-rank nature

$$\Delta\mathbf{W} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top,$$

To minimize quant error

$$\hat{\mathbf{W}} = \mathrm{Quant}_k(\mathbf{W}, \mathbf{X}) = \underset{\hat{\mathbf{W}}}{\mathrm{argmin}}\, ||\mathbf{W}\mathbf{X} - \hat{\mathbf{W}}\mathbf{X}||^2,$$

Based on long-tail distribution distribution, perform mixed-quant

$$\hat{\mathbf{V}}[:, r_{\mathrm{begin}} : r_{\mathrm{end}}]^\top = \mathrm{Quant}_k(\mathbf{V}[:, r_{\mathrm{begin}} : r_{\mathrm{end}}]^\top, \mathbf{X}),$$
$$\hat{\mathbf{U}}[:, r_{\mathrm{begin}} : r_{\mathrm{end}}] = \mathrm{Quant}_k(\mathbf{U}[:, r_{\mathrm{begin}} : r_{\mathrm{end}}], \boldsymbol{\Sigma}[r_{\mathrm{begin}} : r_{\mathrm{end}}, r_{\mathrm{begin}} : r_{\mathrm{end}}]\hat{\mathbf{V}}[:, r_{\mathrm{begin}} : r_{\mathrm{end}}]^\top \mathbf{X}).$$

Average Bits

$$\frac{h_{\mathrm{out}} + h_{\mathrm{in}}}{h_{\mathrm{out}} h_{\mathrm{in}}} \sum_{i=1}^{3} k^{(i)} (r_{\mathrm{end}}^{(i)} - r_{\mathrm{begin}}^{(i)}).$$

Loss-driven search

Table 2: Comparison of different mixed-precision strategies.

# Precision	Setting	GSM8K
Single	1	45.6
	2	50.6
	3	**51.8**
	4	51.6
	8	47.8
	16	43.3
Double	16 + 3	52.5
	8 + 3	**53.1**
	4 + 3	52.2
	3 + 2	52.3
Triple	16 + 8 + 3	53.2
	8 + 4 + 3	52.2
	8 + 3 + 2	**53.6**

Experimental Result

- Llama-2, Mistral, Llama-3 backbones of 7B and 13B sizes
- Math, code , chat and Multi-modal tasks
- All models share the same setting, illustrating generalizability

Table 3: The performance of different delta-compression methods on 7B aligned models.

Method	α	WizardMath GSM8K	WizardMath MATH	MagicoderS-CL HumanEval	MagicoderS-CL MBPP	Llama-2-chat TruthfulQA	Llama-2-chat SafetyBench	LLaVA-v1.5 GQA	LLaVA-v1.5 TextVQA	Ave.
Backbone	1	11.0	2.9	38.4	47.6	41.7	38.9	n/a	n/a	n/a
Aligned	1	55.2	10.9	70.7	69.2	59.5	44.6	62.0	58.2	53.5
Low-Rank	1/16	43.2	8.0	56.7	65.7	55.4	42.5	57.7	53.3	47.8
BitDelta	1/16	45.6	8.6	57.3	65.9	59.3	41.1	59.7	56.9	49.3
Delta-CoMe	1/16	**53.6**	**10.3**	**67.1**	**67.9**	**59.8**	**47.0**	**61.7**	**58.5**	**53.2**

Table 4: The performance of different delta-compression methods on 13B aligned models.

Method	α	WizardMath GSM8K	WizardMath MATH	MagicoderS-CL HumanEval	MagicoderS-CL MBPP	Llama-2-chat TruthfulQA	Llama-2-chat SafetyBench	LLaVA-v1.5 GQA	LLaVA-v1.5 TextVQA	Ave.
Backbone	1	17.8	3.9	43.3	49.0	55.0	37.3	n/a	n/a	n/a
Aligned	1	63.9	14.0	60.4	66.9	62.7	43.9	63.2	61.3	54.5
Low-Rank	1/16	54.2	9.4	53.0	66.9	62.3	43.7	60.2	58.3	51.0
BitDelta	1/16	54.8	10.6	51.8	64.2	62.6	41.6	60.9	60.3	50.9
Delta-CoMe	1/16	**58.9**	**12.8**	**57.9**	**67.2**	**62.9**	**44.1**	**63.1**	**61.2**	**53.5**

Table 5: Results on other representative backbones. The backbone of OpenChat-3.5-0106 (Wang et al., 2023) is Mistral-7B-v0.1 (Jiang et al., 2023). Both Mistral-7B-v0.1 and Llama-3-8B are widely-used open-source LLMs.

Method	α	OpenChat-3.5-0106 GSM8K	OpenChat-3.5-0106 HumanEval	OpenChat-3.5-0106 TruthfulQA	OpenChat-3.5-0106 SafetyBench	Llama-3-8B-Instruct GSM8K	Llama-3-8B-Instruct HumanEval	Llama-3-8B-Instruct TruthfulQA	Llama-3-8B-Instruct SafetyBench	Ave.
Backbone	1	52.2	28.7	61.0	42.1	44.8	33.5	43.6	43.9	43.7
Aligned	1	77.1	73.2	78.4	61.0	78.5	61.6	68.2	51.6	68.7
Low-Rank	1/16	50.5	52.4	76.9	49.0	68.3	46.3	67.5	51.3	57.8
BitDelta	1/16	70.3	54.9	78.4	50.0	67.6	56.1	68.6	50.2	62.0
Delta-CoMe	1/16	**74.8**	**59.8**	**78.9**	**62.6**	**77.1**	**60.4**	**69.1**	**51.8**	**66.8**

Delta-CoMe vs Delta-Tuning

- Delta-tuning uses downstream data to train backbone, delta-compression uses existing aligned models to enhance the backbone.
- Under similar storage budget when inference, Delta compression outperform conventional delta-tuning significantly

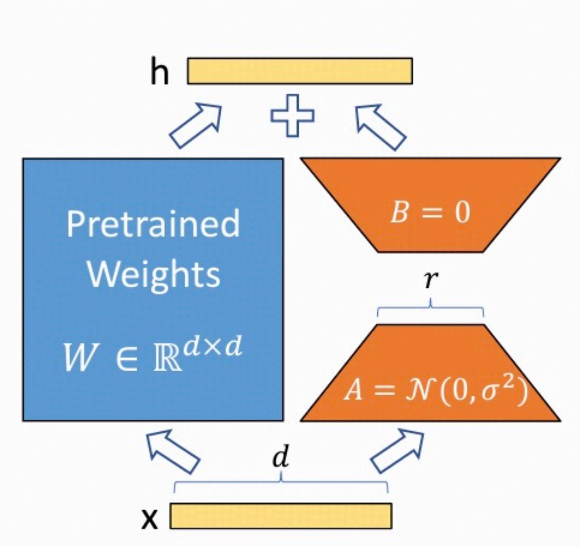


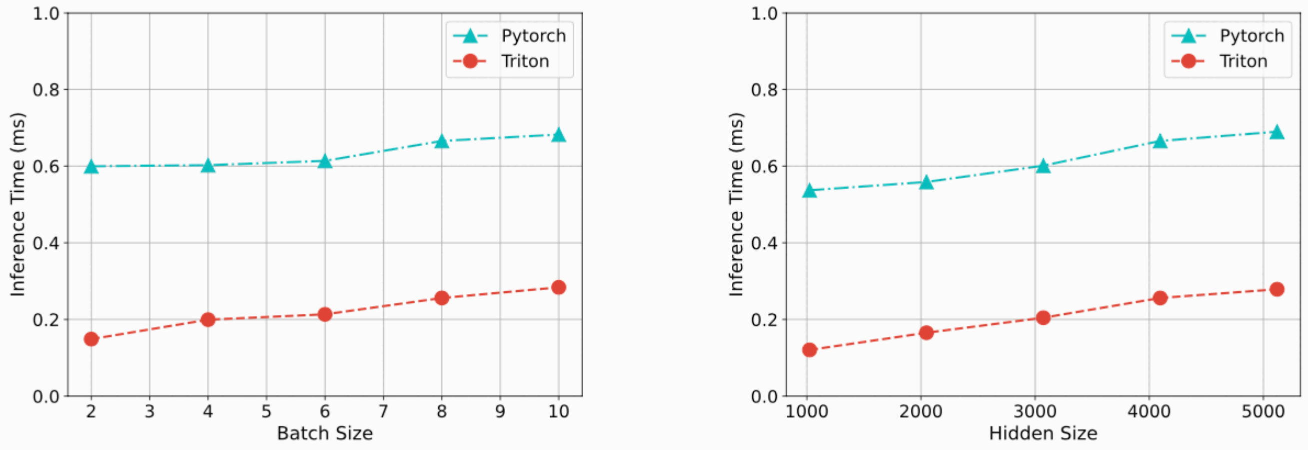
Table 6: Comparison between LoRA and delta-compression methods.

Method	Math		Code		Ave.
	GSM8K	MATH	HumanEval	HuamnEval	
Backbone	11.0	2.9	10.5	17.7	10.5
Aligned	65.4	18.6	43.2	44.9	43.0
LoRA	58.3	11.4	17.6	31.8	29.8
Low-Rank	54.8	5.5	26.2	42.6	32.3
BitDelta	47.8	10.7	26.2	41.9	31.7
Delta-CoMe	**65.1**	**18.0**	**39.6**	**44.9**	**41.9**

Inference Speed and Storage

- Using Triton achieving about 3x speed up than Pytorch
- Saves GPU memory significantly achieving loading 50x models on a single GPU

Num. of Models	w/o DC	w/ DC
2	26.67	15.54
4	52.24	18.17
8	OOM	23.44
16	OOM	33.95
32	OOM	55.06
50	OOM	78.70



(a) Effect of batch size.

(b) Effect of hidden size.

Delta-CoMe with Low-bit Backbone

- Besides the 16-bit backbone, the 4-bit backbone is also widely used.
- Delta-CoMe can also maintain performance with a 4-bit backbone.

Table 10: Performance drop in 4-bit and 16-bit backbone across different tasks.

Precision	Backbone	Tasks	Delta
4-BIT BACKBONE	WizardMath 4-bit	49.36	n/a
	Llama2 4-bit + 1bit delta	47.01	-2.3
16-BIT BACKBONE	WizardMath 16-bit	55.2	n/a
	Llama2 16-bit + 1bit delta	53.6	-1.6
4-BIT BACKBONE	Magicoder 4-bit	66.2	n/a
	Codellama-python 4-bit + 1bit delta	65.4	-0.8
16-BIT BACKBONE	Magicoder 16-bit	66.7	n/a
	Codellama-python 16-bit + 1bit delta	67.2	+0.3
4-BIT BACKBONE	WizardMath 4-bit	49.36	n/a
	Llama2 4-bit + 1bit delta	47.01	-2.3
16-BIT BACKBONE	WizardMath 16-bit	55.2	n/a
	Llama2 16-bit + 1bit delta	53.6	-1.6
4-BIT BACKBONE	Llava-v1.5 4-bit	57.68	n/a
	Vicuna 4-bit + 1bit delta	57.58	-0.1
16-BIT BACKBONE	Llava-v1.5 16-bit	58.2	n/a
	Vicuna 16-bit + 1bit delta	58.5	+0.3

Conclusion

- Delta-CoMe achieves 1-bit compression and near-lossless performance across various typical tasks, including math, code, chat, and multi-modal tasks.
- Delta-CoMe can save more than 10x GPU memory and our kernel has achieved 3x speedup than Pytorch which can be applied into multi-tenant settings.
- However, the kernel is trivial, Wei et al. (2024) and Guo et al. (2024) have implemented more advanced kernels. We can draw on their methods to achieve higher acceleration ratios.