Breaking the Compression Ceiling: Data-Free Pipeline for Ultra-Efficient Delta Compression

Xiaohui Wang[1*] **Peng Ye**[2,3*] **Chenyu Huang**[1] **Shenghe Zheng**[2]
Bo Zhang[2] **Lei Bai**[2] **Wanli Ouyang**[2,3] **Tao Chen**[1,4†]
[1]Fudan University [2]Shanghai AI Laboratory
[3]The Chinese University of Hong Kong [4]Shanghai Innovation Institute
heatherwang000@gmail.com eetchen@fudan.edu.cn

Abstract

With the rise of the fine-tuned–pretrained paradigm, storing numerous fine-tuned models for multi-tasking creates significant storage overhead. Delta compression alleviates this by storing only the pretrained model and the highly compressed delta weights (the differences between fine-tuned and pretrained model weights). However, existing methods fail to maintain both high compression and performance, and often rely on data. To address these challenges, we propose UltraDelta, the first data-free delta compression pipeline that achieves both ultra-high compression and strong performance. UltraDelta is designed to minimize redundancy, maximize information, and stabilize performance across inter-layer, intra-layer, and global dimensions, using three key components: (1) Variance-Based Mixed Sparsity Allocation assigns sparsity based on variance, giving lower sparsity to high-variance layers to preserve inter-layer information. (2) Distribution-Aware Compression applies uniform quantization and then groups parameters by value, followed by group-wise pruning, to better preserve intra-layer distribution. (3) Trace-Norm-Guided Rescaling uses the trace norm of delta weights to estimate a global rescaling factor, improving model stability under higher compression. Extensive experiments across (a) large language models (fine-tuned on LLaMA-2 7B and 13B) with up to 50× compression, (b) general NLP models (RoBERTa-base, T5-base) with up to 224× compression, (c) vision models (ViT-B/32, ViT-L/14) with up to 132× compression, and (d) multi-modal models (BEiT-3) with 18× compression, demonstrate that UltraDelta consistently outperforms existing methods, especially under ultra-high compression. Code is available at https://github.com/xiaohuiwang000/UltraDelta.

1 Introduction

As fine-tuning pretrained models for downstream tasks becomes increasingly popular, a growing number of task-specific fine-tuned models have been developed across various domains. To obtain multi-task capabilities and achieve optimal performance across tasks, deploying multiple fine-tuned models simultaneously has become a common practice. However, multi-model deployment introduces severe storage and computational overhead, since each fine-tuned model requires storing a full set of parameters. Delta Compression has emerged as a promising solution to this problem by substantially reducing storage requirements. Instead of storing multiple complete fine-tuned models, delta compression works by storing a single pretrained model along with a set of delta weights (i.e., the differences between each fine-tuned model and the pretrained model) and then applying aggressive compression to these delta weights to minimize storage overhead. These delta weights

[*]Equal Contribution.
[†]Corresponding Author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-0782

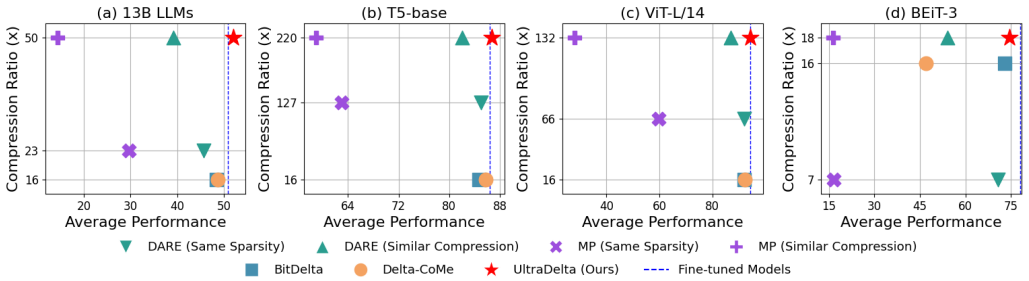

Figure 1: Compression vs. Performance. We compare UltraDelta with baselines, showing strong performance at ultra-high compression. DARE and Magnitude Pruning are evaluated in two settings: with the same sparsity as UltraDelta, and using pruning alone to match a similar compression ratio as UltraDelta. Subfigures show average performance on: (a) LLaMA-2-13B fine-tuned on 3 tasks, (b) T5-base on 8 NLP tasks, (c) ViT-L/14 on 8 vision tasks, and (d) BEiT-3 on 3 vision-language tasks.

are often highly redundant, allowing for substantial compression with minimal impact on model performance, thereby significantly reducing storage costs.

Currently, a number of works have been proposed for delta compression, which can be broadly categorized into two main types: Pruning and Quantization. Pruning methods reduce model size by removing parameters. For example, DARE [91] randomly prunes delta parameters, while Magnitude Pruning [27, 28, 46, 45] removes low-magnitude parameters. Quantization methods work by mapping full-precision parameters to low-bit representations. For example, BitDelta [53] uses masking and rescaling to achieve 1-bit quantization, while Delta-CoMe [64] applies mixed-precision quantization to matrices obtained from singular value decomposition. Other approaches, such as DeltaZip [90], combine quantization with pruning to improve deployment efficiency. However, these methods all struggle to balance ultra-high compression with strong performance, mainly due to: (1) Ignoring inter-layer differences: All layers are treated equally, ignoring that different layers contribute unequally to the model, which limits information preservation. (2) Disrupting intra-layer distributions: Intra-layer weight distributions are crucial to performance but are often distorted by aggressive quantization or pruning. (3) Lack of stability under ultra-high compression: Existing methods struggle to maintain stability under ultra-high compression without relying on data, leading to severe performance drops.

To address these limitations, we propose UltraDelta, the first data-free pipeline to achieve ultra-efficient delta compression, delivering both ultra-high compression and strong performance. UltraDelta focuses on minimizing parameter redundancy, maximizing information preservation, and enhancing model stability across three dimensions: inter-layer, intra-layer and global. It consists of three key components: (1) Variance-Based Mixed Sparsity Allocation (MSA, inter-layer): We theoretically show that layer-wise variance reflects the amount of information in each layer. Based on this insight, we assign lower sparsity to layers with higher variance to better preserve critical information. (2) Distribution-Aware Compression (DAC, intra-layer): We first apply uniform quantization, and group parameters by their quantized value, then perform random pruning within each group to maintain the relative proportions across different values and better preserve the original distribution. (3) Trace-Norm-Guided Rescaling (TNGR, global): We observe that under extreme sparsity, the standard rescaling factor $1/(1-s)$ becomes insufficient to stabilize performance (where s is the sparsity rate). We introduce a refined rescaling factor $\gamma/(1-s)$, where γ is heuristically estimated from the trace norm of each delta weight and is smaller for delta weights with larger trace norms, enhancing robustness under ultra-high compression.

We conduct extensive experiments across models of different scales, types, and tasks to evaluate the effectiveness and robustness of UltraDelta. The results demonstrate its exceptional performance under ultra-high compression: (1) Large Language Models: UltraDelta compresses LLaMA-2 series models [78, 56, 57] to 32× compression for 7B and 50× for 13B, consistently outperforming all baselines and even surpassing the average performance of fine-tuned models. To further demonstrate generality, we also evaluate on more recent architectures, including LLaMA-3.1 [43], Qwen2.5 [89], and Qwen3 [77]. (2) General NLP Models: UltraDelta achieves 224× compression on T5-base [67] and 32× on RoBERTa-base [54] across 8 NLP tasks, even exceeding the average performance of fine-tuned models on T5-base. (3) Vision Models: UltraDelta achieves 50× compression on ViT-B/32 and 132× on ViT-L/14 [66] across 8 image classification tasks, achieving completely lossless performance on ViT-L/14 compared with fine-tuned models. (4) Multi-modal Models: On

BEiT-3 [82], we achieve 18× compression on 3 vision-language tasks, outperforming all baselines across all tasks. As illustrated in Fig. 1, we present compression–performance trade-off for all four model types. UltraDelta consistently achieves the best trade-off, outperforming all baselines, and in some cases, even surpassing the performance of fine-tuned models under ultra-high compression.

The main contributions of this paper can be summarized as follows:

• To break the compression ceiling of delta weights, we analyze the limitations of existing methods in information preservation and model stability, and propose UltraDelta, the first data-free pipeline enabling ultra-efficient delta compression, achieving both ultra-high compression ratios and strong performance without relying on any data.

• To enhance information preservation and model stability, we introduce three novel techniques: Variance-Based Mixed Sparsity Allocation to prioritize critical layers, Distribution-Aware Compression to preserve the original weight distribution, and Trace-Norm-Guided Rescaling to stabilize model performance under extreme sparsity.

• Extensive experiments across large language models, general NLP models, vision models, and multi-modal models demonstrate that UltraDelta consistently outperforms all baselines and even surpasses fine-tuned models in several settings, showcasing its effectiveness and robustness in achieving ultra-efficient delta compression.

2 Related Work

Delta Compression is a technique that compresses the delta weights. It is especially beneficial in multi-model deployment scenarios, where many task-specific models are fine-tuned from the same pretrained model. By storing only the highly compressed delta weights alongside the shared pretrained model, delta compression significantly eliminates redundant storage. Existing methods for delta compression primarily fall into two categories: pruning and quantization.

Delta Weight Pruning reduces model size by removing parameters from the delta weights. Magnitude Pruning (MP)[27, 28, 45, 46] removes weights with the smallest magnitudes but fails to preserve information and causes significant performance drops under high sparsity, as it disrupts the original weight distribution. DARE[91] applies random pruning combined with global rescaling based on sparsity to improve robustness, but suffers from instability under high sparsity. DAREx [20] refines DARE's rescaling using activation statistics, and DeltaDQ [38] performs group-wise pruning and selects the optimal group size based on attention error. However, both methods require data for tuning, limiting their use in data-free scenarios. Some methods also adopt layer-wise pruning [47, 55], but the metrics are often computationally expensive and usually require data. Overall, pruning effectively reduces parameter redundancy but suffers from several drawbacks, such as insufficient information preservation, instability under high sparsity, and reliance on additional data.

Delta Weight Quantization compresses delta weights by mapping them to low-bit representations. GPT-Zip[35] extends GPTQ[22] and quantizes delta parameters to 2-bit. Delta-DCT [33] leverages the discrete cosine transform and compresses delta weights in the frequency domain. BitDelta [53] uses binary masks and scaling factors for 1-bit quantization, but requires calibration data and also distorts the original weight distribution due to the binary masks. Delta-CoMe [64] applies singular value decomposition and mixed-precision quantization on the decomposed matrices to achieve 1-bit compression. Overall, although quantization effectively reduces parameter bit-width, it is often limited to 1-bit precision, which prevents further minimization of parameter redundancy.

Hybrid Approach combines delta weight pruning and delta weight quantization to leverage the advantages of both. DeltaZip [90] employs structured pruning and quantization, primarily aiming to improve deployment efficiency and hardware acceleration, but it falls short in preserving critical information, limiting its effectiveness under ultra-high compression. ComPEFT [88], on the other hand, applies magnitude pruning and then quantizes the remaining weights with a fixed shared value. However, it requires validation data and, due to the reliance on magnitude pruning, disrupts structural integrity and fails to preserve critical information under high sparsity.

In contrast, UltraDelta is a data-free hybrid compression method that explicitly focuses on minimizing parameter redundancy, maximizing information preservation, and enhancing model stability under ultra-high compression ratios, all without requiring any data.

 https://doi.org/10.52202/085713-0782

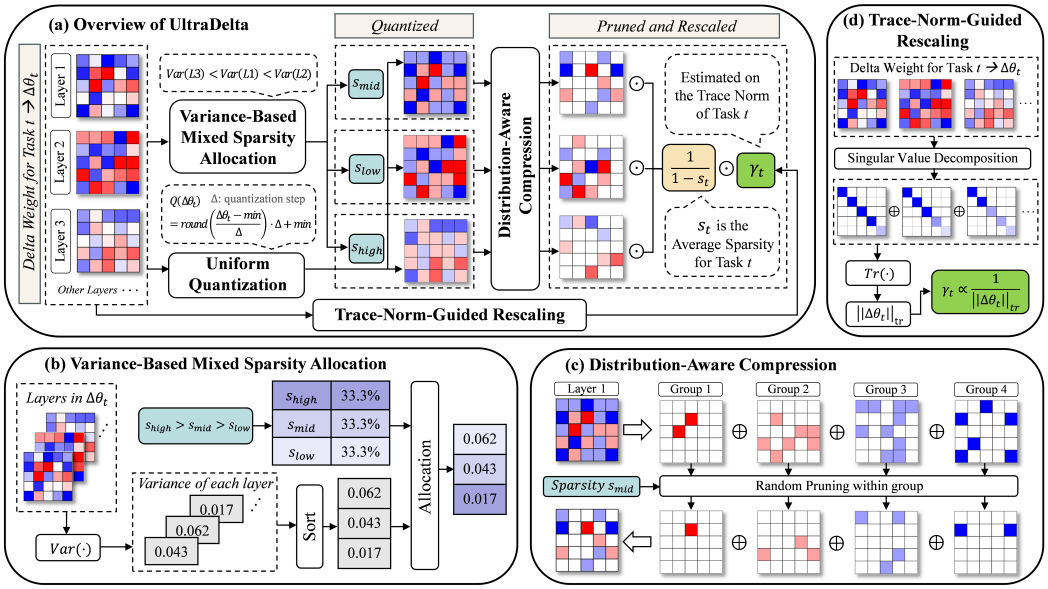

Figure 2: (a) Overview of the UltraDelta pipeline. It first assigns layer-wise sparsity based on variance, and performs uniform quantization followed by group-wise pruning, and finally applies trace-norm-guided rescaling. (b) Variance-Based Mixed Sparsity Allocation allocates lower sparsity to high-variance layers to preserve information. (c) Distribution-Aware Compression groups quantized parameters by value and prunes within each group to retain distribution. (d) Trace-Norm-Guided Rescaling estimates a rescaling factor using the trace norm of each delta weight to enhance robustness.

3 Method

3.1 Preliminaries

Consider a set of T fine-tuned models that are fine-tuned from the same pretrained model. Let the weights of these models be denoted as $\{\theta_1, \theta_2, \ldots, \theta_T\}$, and the shared pretrained model weight as θ_{pre}. For each fine-tuned model $t \in \{1, 2, \ldots, T\}$, we define the delta weight as:

$$\Delta\theta_t = \theta_t - \theta_{\text{pre}} \tag{1}$$

3.2 UltraDelta: A Data-Free Pipeline for Ultra-Efficient Delta Compression

To address the limitations of existing delta compression methods, UltraDelta introduces a hybrid compression framework that minimizes parameter redundancy while maximizing information preservation and enhancing model stability across three dimensions: inter-layer, intra-layer, and global, enabling robustness under ultra-high compression. An overview of the overall pipeline is illustrated in Fig. 2. We detail the core design of UltraDelta in Sec.3.2, and provide analysis in Sec.3.3.

(1) Variance-Based Mixed Sparsity Allocation (MSA). Existing pruning methods often adopt uniform sparsity across layers, treating all parts of the model equally and overlooking their varying importance. However, different layers exhibit different sensitivities to pruning. We observe that layers with higher variance tend to carry more critical information and are more sensitive to pruning (see Sec. 3.3). Motivated by this, we propose MSA that categorizes all layers into three groups based on their variances (low, medium, high), each containing an equal number of parameters:

$$\text{Groups} = \{\mathcal{G}_{\text{low}}, \mathcal{G}_{\text{mid}}, \mathcal{G}_{\text{high}}\} \tag{2}$$

Each group $\mathcal{G}_v$ is assigned a sparsity rate s_v based on its variance $v \in \{\text{low}, \text{mid}, \text{high}\}$, with lower-variance groups assigned higher sparsity and higher-variance groups assigned lower sparsity, in order to preserve inter-layer information better:

$$s_v = s_{\text{mid}} + \delta_v, \quad \delta_v \in \{+s_{\text{step}}, 0, -s_{\text{step}}\} \tag{3}$$

where s_{mid} is the average sparsity of the overall delta weight, and $s_{\text{step}} > 0$ controls the difference in sparsity across groups.

https://doi.org/10.52202/085713-0782

(2) Distribution-Aware Compression (DAC). Prior work has shown that preserving the shape of the weight distribution after compression helps maintain model performance [33]. To better preserve the intra-layer distribution shape under ultra-high compression, we propose DAC. Specifically, as delta weight exhibits a distribution well-suited to uniform quantization [38], we first apply uniform quantization with a lower bit width b (typically $b = 4$) to the delta weight. This quantization process consists of two steps: (a) mapping each value in $\Delta\theta_t$ to a discrete integer within $[0, 2^b - 1]$; (b) mapping the quantized values back to the original value range. Let $\min = \min(\Delta\theta_t)$, $\max = \max(\Delta\theta_t)$, and Δ be the quantization step size:

$$\hat{\Delta\theta}_t = \text{round}\left(\frac{\Delta\theta_t - \min}{\Delta}\right) \cdot \Delta + \min, \quad \text{where} \quad \Delta = \frac{\max - \min}{2^b - 1} \tag{4}$$

After quantization, we group the parameters by their values and perform random pruning within each group. Let $\{u_1, u_2, \ldots, u_n\}$ represent the unique values in the quantized delta weight $\hat{\Delta\theta}_t$. We denote (j, k) as the row and column indices of elements in $\hat{\Delta\theta}_t$. For each unique quantized value u_i, we define a mask $\boldsymbol{M}_{u_i}$ whose elements are independently sampled from a Bernoulli distribution with success probability $1 - s_l$, where s_l is the sparsity rate of the layer. Specifically:

$$\boldsymbol{M}_{u_i}^{(j,k)} \sim \begin{cases} \text{Bernoulli}(1 - s_l), & \text{if } \hat{\Delta\theta}_t^{(j,k)} = u_i \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

The pruned delta weight $\hat{\Delta\theta}_t^*$ is obtained by applying the group-wise Bernoulli masks to the quantized delta weights:

$$\hat{\Delta\theta}_t^* = \sum_{i=1}^{n} u_i \cdot \boldsymbol{M}_{u_i} \tag{6}$$

DAC preserves the relative proportions of delta values, maintaining the intra-layer distribution shape. It minimizes distortion under high sparsity and ensures more stable performance. Moreover, DAC can be applied in non-quantized settings with minimal modification, making it suitable for scenarios where aggressive compression is not required. Instead of grouping by discrete values, we partition the weight range into several intervals, and assign parameters whose values fall within the same interval to the same group. Details are provided in App. A.

(3) Trace-Norm-Guided Rescaling (TNGR). As derived in Sec. 3.3, the variance of activation errors between compressed and original delta weights correlates with $s/(1 - s)$, causing instability at high sparsity. To address this, we introduce an additional rescaling factor γ_t for task t, and redefine rescaling as $\gamma_t/(1 - s_t)$, where s_t is the overall sparsity of the delta weight for task t. We find that delta weights with larger trace norms (i.e., the sum of their singular values) tend to require smaller γ_t to maintain stable performance. Motivated by this observation, we set γ_t inversely proportional to the trace norm of the delta weights:

$$\gamma_t \propto \frac{1}{||\Delta\theta_t||_{\text{tr}}} \tag{7}$$

where $\|\cdot\|_{\text{tr}}$ denotes the trace norm. This trace-norm-guided rescaling offers a simple yet effective way to adaptively stabilize pruning under extreme sparsity. In practice, γ_t typically falls within the range of $[0.5, 1.0]$. During the inference stage, the final model weight is reconstructed as:

$$\theta_t^{\text{final}} = \theta_{\text{pre}} + \frac{\gamma_t}{1 - s_t} \cdot \hat{\Delta\theta}_t^* \tag{8}$$

3.3 Reason for Effectiveness

3.3.1 Theoretical Analysis of MSA

In lossless compression, the entropy represents the lower bound on the achievable average bit rate. A larger entropy means that a layer carries more information and therefore should be more carefully preserved. We show that the variance of a layer is closely related to its entropy, serving as the theoretical motivation for MSA. Following [50], we model the distribution of a layer as a Gaussian Distribution $L \sim \mathcal{N}(\mu, \sigma^2)$. The probability density function of L is given by:

$$p(l) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(l - \mu)^2}{2\sigma^2}\right) \tag{9}$$

https://doi.org/10.52202/085713-0782

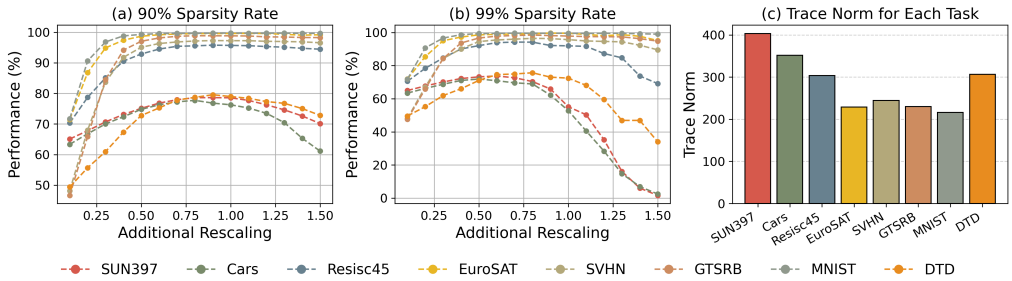

Figure 3: Relationship between Trace Norm and Additional Rescaling Factor on 8 ViT-B/32. (a) Average performance under 90% sparsity with different additional rescaling factors. (b) Same as (a), under 99% sparsity. (c) Trace norm of delta weights across tasks.

The entropy of this distribution is (detailed derivation in App. B.1):

$$H(L) = -\mathbb{E}[\log p(l)] = \log(\sigma) + \frac{1}{2}\log(2\pi) + \frac{1}{2} \tag{10}$$

This demonstrates the link between variance and entropy: layers with larger variance have higher entropy, implying greater information content and thus requiring smaller sparsity to preserve it.

In lossy compression, entropy alone is not sufficient. When distortion is allowed, we employ rate–distortion theory [6, 17], which establishes the limit on how much information must be retained to achieve a given distortion level. Let $\hat{L}$ denote the compressed version of L. With mean squared error (MSE) distortion, we define:

$$D = \mathbb{E}\big[(L - \hat{L})^2\big], \tag{11}$$

According to rate–distortion theory, the rate–distortion function of a Gaussian source under MSE distortion (which is independent of the mean μ) is given by:

$$R(D) = \begin{cases} \frac{1}{2}\log\left(\frac{\sigma^2}{D}\right), & 0 < D < \sigma^2, \\ 0, & D \geq \sigma^2, \end{cases} \tag{12}$$

This shows that, for a fixed distortion D, a larger variance σ^2 leads to a higher rate $R(D)$, meaning that more information must be retained. In other words, layers with larger variance require a smaller sparsity in order to preserve their information under lossy compression.

3.3.2 Theoretical Analysis of the Rescaling Factor.

For a given delta weight $\Delta\theta$, let the activation be defined as $a = \Delta\theta \odot x$, where x is the input feature. We introduce a Bernoulli mask $B \sim \text{Bernoulli}(1-s)$, with sparsity rate s, and apply an additional scaling factor $\gamma \in [0, 1]$. Following [20], the activation error introduced by compression is:

$$\varepsilon = \Delta\theta \odot x - \frac{\gamma}{1-s} \cdot (B \odot \Delta\theta \odot x) = a \odot \left(1 - B \cdot \frac{\gamma}{1-s}\right) \tag{13}$$

We further derive the variance of this error (detailed derivation in App. B.2):

$$\text{Var}(\varepsilon) = \frac{\gamma^2 s}{1-s} \odot a^2 \tag{14}$$

As sparsity becomes higher, the variance grows rapidly and causes instability. This underscores the importance of using smaller rescaling factors to stabilize the model under high sparsity.

3.3.3 Empirical Analysis of TNGR.

We investigate how the additional rescaling factor γ for each delta weight correlates with its intrinsic characteristics, as shown in Fig. 3. At 90% sparsity, we observe that delta weights with larger trace norms tend to require smaller values of γ to maintain stable performance. Moreover, their performance is more sensitive to changes in γ. This phenomenon becomes even more pronounced at 99% sparsity, where the need for smaller γ is more substantial for delta weights with large trace norms, and the performance drop-off becomes steeper when γ is not well matched. These empirical trends support our trace-norm-guided heuristic estimation, which sets $\gamma \in [0.5, 1.0]$ inversely proportional to the trace norm for adaptive and data-free rescaling.

3.3.4 Ability to Preserve Information.

To explain why UltraDelta achieves strong performance under ultra-high compression ratios, we analyze how well it preserves the original model's internal representations. Specifically, we compare the cosine similarity between the embeddings from the compressed and fine-tuned RoBERTa-base [54] models on the CoLA [83] dataset. We extract the layer-wise embeddings of each input token and report the average cosine similarities. As shown in Fig. 4, our method consistently maintains high similarity across all layers, especially in the deeper ones that are more critical to performance. The distribution of cosine similarities in the final layer further confirms that UltraDelta retains essential information, indicating its effectiveness in preserving critical information despite ultra-high compression.

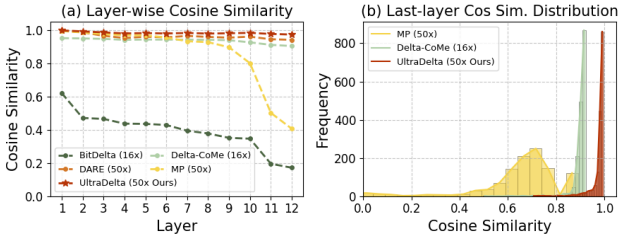

Figure 4: Cosine Similarity Analysis. (a) Layer-wise similarity of each layer's embeddings. (b) Distribution of cosine similarity for the final-layer embeddings. Compression ratios for each method are indicated in parentheses.

4 Experiment

Baseline Methods. We compare UltraDelta with the following baselines: (1) Fine-tuned Models; (2) Multi-task Learning (MTL) [8], which trains a single model on all tasks; (3) BitDelta [53]; (4) Delta-CoMe [64]; (5) DARE [91] with either the same sparsity or similar compression ratio as UltraDelta, using pruning alone to achieve a similar compression ratio; (6) Magnitude Pruning (MP) [27], also matched by sparsity or compression ratio. Since BitDelta and Delta-CoMe target only linear layers in Transformer blocks, we restrict all methods to compress only linear layers for fairness. All original models are stored in FP16 format. Experiments are conducted on a NVIDIA A800 GPU.

Compression Ratio. Unlike quantization, where compression is directly determined by bit-width, pruning methods require storing the indices of non-zero values. To ensure a unified and fair evaluation, following [70, 74, 88], we adopt Golomb coding [25] to store the compressed delta weights for methods involving pruning, which is well-suited for encoding the zero-run lengths that approximately follow a geometric distribution. Detailed derivation and calculations are in App. B.3, with results summarized in Tab. 8. Furthermore, for a complete evaluation, we present the ideal upper bound of compression ratio and practical results under different storage schemes in Sec. 4.5(storage scheme).

4.1 Performance on Large Language Models (LLMs)

Settings. We primarily evaluate UltraDelta on the LLaMA-2 series with sizes of 7B and 13B across three types of models: math (WizardMath [56]), code (WizardCoder [57]), and chat (LLAMA-2-Chat [78]). The models are evaluated using GSM8K [15] for math (accuracy), HumanEval [11] for code (pass@1), and TruthfulQA [49] for chat (accuracy). We also incorporate more recent LLMs and more challenging tasks: LLaMA-3.1-Tulu-8B [43] evaluated on MBPP+ [3] and HumanEval+ [11], Qwen2.5-7B-Instruct [89] evaluated on MATH [31] and GPQA [69], and Qwen3Guard-8B [77] evaluated on MMLU [30] and BBQ [63]. Details for the LLM IDs are presented in App. C.1.

Results. We first report results on the LLaMA-2 series (see Tab. 1). For the 7B model, we apply 4-bit quantization and prune 95% of parameters, achieving an 32.9× compression ratio; for the 13B model, we prune 97% to reach 50.9× compression. Despite these ultra-high compression ratios, UltraDelta achieves average scores of 45.57 (7B) and 52.05 (13B), exceeding the fine-tuned models (45.37 and 50.94, respectively). This improvement suggests that UltraDelta may introduce a regularization benefit. Compared to BitDelta [53] and Delta-CoMe [64], which are limited to 16× compression, UltraDelta delivers both higher compression and better performance. Other approaches such as DARE [91] and MP [27] fall short even at lower compression ratios, and when matched to the similar compression ratio as UltraDelta, their performance drops sharply. For newer LLMs (see Tab. 2), UltraDelta achieves the best overall compression–performance trade-off across baselines. In summary, UltraDelta consistently outperforms all baselines across various tasks and model sizes, demonstrating its remarkable robustness and effectiveness under ultra-high compression.

https://doi.org/10.52202/085713-0782

Table 1: Performance comparison on large language models across 3 tasks. Ratio denotes the compression ratio (original model size / compressed size). Methods marked with "(Sparsity)" use the same sparsity rate as ours, while those marked with "(Compression)" use a higher sparsity rate to match a similar overall compression ratio as our method. Best results are highlighted in bold.

Method	Ratio		WizardMath-V1.0		WizardCoder-V1.0		LLAMA-2-Chat		Avg	
	7B	13B	7B	13B	7B	13B	7B	13B	7B	13B
Individual	1×	1×	41.55	47.31	49.40	58.50	45.17	47	45.37	50.94
BitDelta [53]	16×	16×	36.54	46.12	46.30	54.30	42.84	44.85	41.89	48.42
Delta-CoMe [64]	16×	16×	37.25	46.87	45.30	53.70	45.02	45.57	42.52	48.71
DARE [91] (Sparsity)	14.7×	23.7×	36.62	46.70	46.20	53.90	33.54	36.29	38.79	45.63
MP [27] (Sparsity)	14.7×	23.7×	23.65	3.11	45.70	47.60	40.02	38.25	36.46	29.65
DARE [91] (Compression)	31.7×	50.1×	29.49	35.63	43.30	51.80	29.13	33.25	33.97	40.23
MP [27] (Compression)	31.7×	50.1×	16.22	8.42	32.90	16.50	31.95	21.15	27.02	15.36
UltraDelta (Ours)	**32.9×**	**50.9×**	**38.76**	**48.45**	**51.80**	**59.10**	**46.14**	**48.59**	**45.57**	**52.05**

Table 2: Performance comparison on recent large language models.

Method	Ratio	Llama-3.1-Tulu-3-8B		Qwen2.5-7B-Instruct		Qwen3Guard-Gen-8B		Avg
		HumanEval+	MBPP+	MATH	GPQA	MMLU	BBQ	
Individual	1×	57.30	56.60	36.20	36.87	72.80	50.34	51.69
BitDelta [53]	16×	55.50	52.90	38.10	37.37	73.05	49.60	51.09
Delta-CoMe [64]	16×	53.80	53.10	29.45	31.43	71.46	44.61	47.31
DARE [91] (Sparsity)	14.7×	50.60	50.00	37.76	34.85	72.18	49.69	49.18
MP [27] (Sparsity)	14.7×	45.10	40.20	11.84	31.82	70.91	**50.18**	41.68
UltraDelta (Ours)	**32.9×**	**56.10**	**54.50**	**40.14**	**38.38**	**73.44**	49.74	**52.05**

4.2 Performance on general NLP models

Settings. Following [76, 91], we evaluate both T5-base [67] and RoBERTa-base [54] models on the GLUE [81] benchmark, covering CoLA [83], SST-2 [71], MRPC [21], STS-B [10], QQP [36], MNLI [84], QNLI [68], and RTE [24]. For T5-base, we report Spearman's ρ on STS-B and accuracy on the other tasks. Settings and results of RoBERTa-base are provided in App. C.3.

Results. The results for T5-base models are presented in Tab. 3. Using 4-bit quantization combined with 99.5% pruning, UltraDelta achieves an impressive 224.6× compression ratio. Despite this extreme compression, UltraDelta attains an average accuracy of 86.74, outperforming all existing compression baselines and even slightly exceeding the full fine-tuned model's accuracy of 86.37. Compared to BitDelta [53] and Delta-CoMe [64], UltraDelta achieves a much higher compression ratio while still maintaining superior performance. This demonstrates that UltraDelta successfully pushes the compression limits of delta weights while preserving model performance, highlighting its effectiveness and strong generalization ability.

Table 3: Performance comparison on T5-base models across 8 tasks.

Methods	Ratio	CoLA	SST2	MRPC	STSB	QQP	MNLI	QNLI	RTE	Avg
Individual	1×	74.98	93.58	87.50	88.70	85.37	83.41	91.49	85.92	86.37
BitDelta [53]	16×	70.09	93.46	84.06	86.20	85.26	**83.64**	90.94	83.75	84.68
Delta-CoMe [64]	16×	74.26	93.58	87.01	87.94	85.44	83.51	91.54	84.36	85.95
DARE [91] (Sparsity)	127.6×	74.16	93.34	87.83	87.86	85.22	83.19	91.52	84.11	85.90
MP [27] (Sparsity)	127.6×	37.87	88.76	70.59	0	80.4	68.55	82.88	82.67	63.97
DARE [91] (Compression)	220.5×	71.57	92.25	85.03	86.77	84.26	81.16	90.50	82.81	84.30
MP [27] (Compression)	220.5×	30.87	87.84	70.83	0	79.16	56.36	76.31	82.17	60.44
UltraDelta (Ours)	**224.6×**	**76.51**	**93.81**	**88.48**	**88.82**	**85.41**	83.17	**91.76**	**85.92**	**86.74**

4.3 Performance on Vision Models

Settings. Following [34], we evaluate ViT-B/32 and ViT-L/14 [66] models on eight image classification datasets: SUN397 [87], Stanford Cars [40], RESISC45 [12], EuroSAT [29], SVHN [60], GTSRB [73], MNIST [44], and DTD [13]. Accuracy is used as the evaluation metric for all datasets.

Results. The results of ViT-L/14 models are shown in Tab. 4, while ViT-B/32 results are provided in App. C.2. With 4-bit quantization and 99% sparsity, UltraDelta achieves a 132.5× compression ratio while maintaining an average accuracy of 94.4, matching the full fine-tuned model and significantly outperforming all compressed baselines, demonstrating its effectiveness on vision models.

Table 4: Performance comparison on ViT-L/14 models across 8 tasks.

Methods	Ratio	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg
Individual	1×	84.8	92.3	97.4	99.7	98.1	99.2	99.7	84.1	94.4
Traditional MTL [8]	8×	80.8	90.6	96.3	96.3	97.6	99.1	99.6	84.4	93.1
BitDelta [53]	16×	84.0	92.1	97.2	99.7	97.9	99.0	99.7	83.0	94.1
Delta-CoMe [64]	16×	**84.3**	92.1	97.2	99.6	98.0	99.1	99.7	83.6	94.2
DARE [91] (Sparsity)	66.5×	82.2	90.0	97.0	99.7	98.0	99.1	99.7	83.1	93.6
MP [27] (Sparsity)	66.5×	26.7	43.9	83.4	92.8	24.4	62.9	77.7	66.1	59.7
DARE [91] (Compression)	127.6×	65.2	78.4	94.2	99.6	97.7	99.2	99.7	79.1	89.1
MP [27] (Compression)	127.6×	4.9	4.2	42.6	83.2	6.4	16.7	27.2	50.7	29.5
UltraDelta (Ours)	**132.5**×	84.1	**92.2**	**97.4**	**99.8**	**98.1**	**99.2**	**99.8**	**84.3**	**94.4**

4.4 Performance on multi-modal models

Settings. We compress delta weights on BEiT-3 [82] models fine-tuned on three datasets: VQA [26] (Visual Question Answering), NLVR2 [75] (Visual Reasoning), and COCO Captioning [51] (Image Captioning). The COCO Captioning task is evaluated using BLEU4 [61], CIDEr [79], METEOR [4], and ROUGE-L [48], while the other two tasks are evaluated based on accuracy.

Results. The results of BEiT-3 models are shown in Tab. 5. With 4-bit quantization and 90% sparsity, UltraDelta achieves a 18.4× compression ratio maintaining high accuracy and quality, outperforming all baselines across all tasks, demonstrating its effectiveness on multi-modal models.

Table 5: Performance comparison on multi-modal BEiT-3 models across 3 vision-language tasks.

Methods	Task	Ratio	COCO-Captioning				NLVR2	VQAv2
	Metric	$\alpha(\uparrow)$	BLEU4(↑)	CIDEr(↑)	METEOR(↑)	ROUGE-L(↑)	Accuracy(↑)	Accuracy(↑)
Individual		1×	0.394	1.337	0.331	0.601	84.269	83.49
BitDelta [53]		16×	0.367	1.250	0.294	0.581	82.317	74.59
Delta-CoMe [64]		16×	0.224	0.846	0.251	0.488	60.098	38.64
DARE [91] (Sparsity)		7.7×	0.333	1.15	0.285	0.564	80.652	73.29
MP [27] (Sparsity)		7.7×	0	0	0.006	0.014	49.263	0
DARE [91] (Compression)		18.1×	0.178	0.589	0.196	0.441	69.470	58.89
MP [27] (Compression)		18.1×	0	0	0.006	0.013	48.931	0.01
UltraDelta (Ours)		**18.4**×	**0.372**	**1.270**	**0.296**	**0.583**	**83.163**	**75.66**

4.5 Ablation Study

Effectiveness of Each Component. We evaluate the contribution of each component on 8 ViT-B/32 in Tab. 6. DAC substantially improves compression ratio from 23.7× to 50.9×, all while preserving model accuracy. MSA yields the largest accuracy improvement, highlighting the effectiveness of mixed sparsity allocation. Together, they enable ultra-high compression with strong performance.

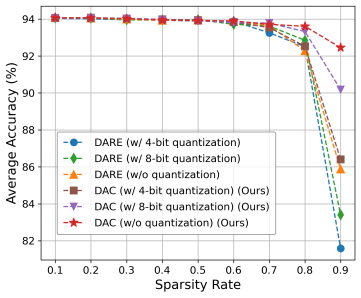

Figure 5: Average Performance of DAC and DARE on 30 ViT-B/16.

Effectiveness of DAC. To further validate the robustness of DAC, we evaluate it on a large-scale benchmark [32] of 30 ViT-B/16 models across 4-bit, 8-bit, and non-quantized configurations. For fair comparison, we apply uniform quantization to both methods in quantized cases. As shown in Fig. 5 (detailed results are in App. C.4.1), DAC consistently outperforms DARE [91] across all configurations. Notably, under 90% sparsity and 4-bit quantization, DAC still exceeds DARE's non-quantized performance, confirming its robustness across diverse tasks.

https://doi.org/10.52202/085713-0782

Table 6: Ablation study on the effectiveness of each component.

Methods	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Ratio	Avg
DARE [91] (97% Sparsity)	76.0	73.3	95.1	99.7	97.0	98.5	99.6	78.1	23.7×	89.7
+ DAC	75.6	73.3	95.3	99.5	97.1	98.5	99.6	78.5	50.9× (↑ 27.2×)	89.7
+ DAC + MSA	76.9	75.9	95.5	99.2	97.0	98.8	99.7	79.2	50.9×	90.3 (↑ 0.6)
+ DAC + MSA + TNGR	78.6	77.4	95.7	99.4	97.0	98.7	99.7	79.0	50.9×	90.7 (↑ 0.4)

Table 7: Compression ratios across storage formats and models.

Format	LLaMA-2-7B	LLaMA-2-13B	ViT-B/32	ViT-L/14	T5-base	RoBERTa-base	BEiT-3
Index-Free Storage	80.0×	133.0×	133.0×	400.0×	800.0×	80.0×	40.0×
Golomb Coding	32.9×	50.9×	50.9×	132.5×	224.6×	32.9×	18.4×
CSR	19.5×	30.7×	35.4×	105.2×	188.3×	21.4×	10.8×
BCSR	12.7×	19.8×	21.7×	63.7×	121.4×	13.4×	7.3×

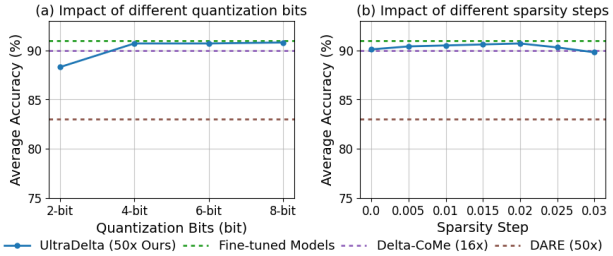

Figure 6: Impact of hyper-parameter on 8 ViT-B/32. (a) Impact of quantization bits; (b) Impact of sparsity steps.

Impact of Hyper-Parameters. We evaluate key hyper-parameters in DAC and MSA on 8 ViT-B/32 models, as shown in Fig. 6 (detailed results are in App. C.4.2). For bit-width in quantization (see Eq. 4), accuracy improves notably from 2-bit to 4-bit, with marginal gains beyond, suggesting 4-bit as an optimal balance. For sparsity step s_{step} in MSA (see Eq. 3), we evaluate it under a fixed 97% target sparsity and 4-bit quantization. A moderate step size (within $[0.01, 0.02]$) yields the best average performance. Performance remains stable as long as extremely large or small step sizes are avoided. This indicates that precise tuning of s_{step} is generally unnecessary.

Storage Scheme. We report compression ratios of UltraDelta under different storage schemes. Index-Free Storage denotes the ideal upper bound where only non-zero values are stored without indices (the ideal compression ratios of other pruning baselines are in Tab. 8). We also evaluate practical schemes, including Golomb coding, Compressed Sparse Row (CSR), and Block Compressed Sparse Row (BCSR). As shown in Tab. 7, BCSR performs poorly, as it only works when zeros are densely clustered and thus is not suitable for our setting.

5 Discussion and Conclusion

Regularization Effect. By controlling the fitting degree of models, we examine that UltraDelta particularly benefits underfitted models (see App. D.1 for detailed results and analysis). Since LLMs are typically evaluated in the zero-shot or few-shot settings, where underfitting is common, this explains why UltraDelta often yields not only strong compression but also performance gains.

Overhead. UltraDelta introduces only modest computational overhead during compression and is efficient at inference compared with baselines. The main cost arises from computing trace norms. Since only their relative magnitudes are required, fast approximation such as randomized SVD can be employed to reduce the overhead. In the inference stage, delta weights only need to be dequantized and added back to the base model, which can be done on CPU efficiently. In contrast, methods such as Delta-CoMe [64] require additional matrix multiplications, making them less efficient.

Limitations and Future Work. We acknowledge that our method employs a heuristic rescaling strategy and does not explicitly target deployment efficiency; detailed discussion is in App. D.3.

Conclusion. In this paper, we present UltraDelta, the first data-free pipeline for ultra-efficient delta compression, capable of achieving both ultra-high compression ratios and strong performance. UltraDelta is a hybrid compression method with a focus on preserving information across inter-layer, intra-layer, and global dimensions through three novel components. Extensive experiments on language, vision, and multi-modal models, along with theoretical analysis, validate the effectiveness, robustness, and generality of UltraDelta across diverse settings.

https://doi.org/10.52202/085713-0782

6 Acknowledgement

This work is supported by National Key Research and Development Program of China (No. 2022ZD0160101), Shanghai Natural Science Foundation (No. 23ZR1402900), Shanghai Science and Technology Commission Explorer Program Project (24TS1401300), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). The computations in this research were performed using the CFFF platform of Fudan University.

References

[1] M Israk Ahmed, Shahriyar Mahmud Mamun, and Asif Uz Zaman Asif. Dcnn-based vegetable image classification using transfer learning: A comparative study. In *2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pages 235–243. IEEE, 2021.

[2] Sarder Iftekhar Ahmed, Muhammad Ibrahim, Md Nadim, Md Mizanur Rahman, Maria Mehjabin Shejunti, Taskeed Jabid, and Md Sawkat Ali. Mangoleafbd: A comprehensive image dataset to classify diseased and healthy mango leaves. *Data in Brief*, 47:108941, 2023.

[3] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[4] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[5] Puneet Bansal. Intel image classification. *Available on https://www. kaggle. com/puneet6060/intel-image-classification, Online*, 2019.

[6] Toby Berger. Rate-distortion theory. *Wiley Encyclopedia of Telecommunications*, 2003.

[7] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[8] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[9] CCHANG. Garbage classification. `https://www.kaggle.com/ds/81794`, 2018.

[10] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

[11] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[12] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

[13] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[14] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[15] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[16] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[17] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[18] Will Cukierski. Dogs vs. cats, 2013. *URL https://kaggle. com/competitions/dogs-vs-cats*.

 https://doi.org/10.52202/085713-0782

[19] DeepNets. Landscape recognition. https://www.kaggle.com/datasets/utkarshsaxenadn/landscape-recognition-image-dataset-12k-images.

[20] Wenlong Deng, Yize Zhao, Vala Vakilian, Minghui Chen, Xiaoxiao Li, and Christos Thrampoulidis. Dare the extreme: Revisiting delta-parameter pruning for fine-tuned models. *arXiv preprint arXiv:2410.09344*, 2024.

[21] Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*, 2005.

[22] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[23] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 2024.

[24] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9, 2007.

[25] Solomon Golomb. Run-length encodings (corresp.). *IEEE transactions on information theory*, 12(3): 399–401, 1966.

[26] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[27] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[28] 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.

[29] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[30] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[31] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[32] Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. Emr-merging: Tuning-free high-performance model merging. *Advances in Neural Information Processing Systems*, 37:122741–122769, 2024.

[33] Chenyu Huang, Peng Ye, Xiaohui Wang, Shenghe Zheng, Biqing Qi, Lei Bai, Wanli Ouyang, and Tao Chen. Seeing delta parameters as jpeg images: Data-free delta compression with discrete cosine transform. *arXiv preprint arXiv:2503.06676*, 2025.

[34] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

[35] Berivan Isik, Hermann Kumbong, Wanyi Ning, Xiaozhe Yao, Sanmi Koyejo, and Ce Zhang. Gpt-zip: Deep compression of finetuned large language models. In *Workshop on Efficient Systems for Foundation Models@ ICML2023*, 2023.

[36] Shankar Iyer, Nikhil Dandekar, Kornél Csernai, et al. First quora dataset release: Question pairs. data. quora. com. 2017.

[37] Mona Jalal, Kaihong Wang, Sankara Jefferson, Yi Zheng, Elaine O Nsoesie, and Margrit Betke. Scraping social media photos posted in kenya and elsewhere to detect and analyze food types. In *Proceedings of the 5th International Workshop on Multimedia Assisted Dietary Management*, pages 50–59, 2019.

[38] Yanfeng Jiang, Zelan Yang, Bohua Chen, Shen Li, Yong Li, and Tao Li. Deltadq: Ultra-high delta compression for fine-tuned llms via group-wise dropout and separate quantization. *arXiv preprint arXiv:2410.08666*, 2024.

[39] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, 2011.

[40] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[41] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[42] Makerere AI Lab. Bean disease dataset, 2020.

[43] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. 2024.

[44] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[45] Jaeho Lee, Sejun Park, Sangwoo Mo, Sungsoo Ahn, and Jinwoo Shin. Layer-adaptive sparsity for the magnitude-based pruning. *arXiv preprint arXiv:2010.07611*, 2020.

[46] Guiying Li, Chao Qian, Chunhui Jiang, Xiaofen Lu, and Ke Tang. Optimization based layer-wise magnitude-based pruning for dnn compression. In *IJCAI*, pages 2383–2389, 2018.

[47] Wei Li, Lujun Li, Mark Lee, and Shengjie Sun. Adaptive layer sparsity for large language models via activation correlation assessment. *Advances in Neural Information Processing Systems*, 37:109350–109380, 2024.

[48] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[49] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

[50] Sihao Lin, Pumeng Lyu, Dongrui Liu, Tao Tang, Xiaodan Liang, Andy Song, and Xiaojun Chang. Mlp can be a good transformer learner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19489–19498, 2024.

[51] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[52] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[53] James Liu, Guangxuan Xiao, Kai Li, Jason D Lee, Song Han, Tri Dao, and Tianle Cai. Bitdelta: Your fine-tune may only be worth one bit. *Advances in Neural Information Processing Systems*, 37:13579–13600, 2024.

[54] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[55] Haiquan Lu, Yefan Zhou, Shiwei Liu, Zhangyang Wang, Michael W Mahoney, and Yaoqing Yang. Alphapruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models. *Advances in neural information processing systems*, 37:9117–9152, 2024.

[56] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

 https://doi.org/10.52202/085713-0782

[57] Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*, 2023.

[58] Alexander Mamaev. Flowers recognition. `https://www.kaggle.com/datasets/alxmamaev/flowers-recognition`.

[59] Horea Muresan and Mihai Oltean. Fruit recognition from images using deep learning. *Acta Universitatis Sapientiae, Informatica*, 10(1):26–42, 2018.

[60] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 4. Granada, 2011.

[61] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[62] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[63] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel R Bowman. Bbq: A hand-built bias benchmark for question answering. *arXiv preprint arXiv:2110.08193*, 2021.

[64] Bowen Ping, Shuo Wang, Hanqing Wang, Xu Han, Yuzhuang Xu, Yukun Yan, Yun Chen, Baobao Chang, Zhiyuan Liu, and Maosong Sun. Delta-come: Training-free delta-compression with mixed-precision for large language models. *Advances in Neural Information Processing Systems*, 37:31056–31077, 2024.

[65] Konstantin Pogorelov, Kristin Ranheim Randel, Carsten Griwodz, Sigrun Losada Eskeland, Thomas de Lange, Dag Johansen, Concetto Spampinato, Duc-Tien Dang-Nguyen, Mathias Lux, Peter Thelin Schmidt, et al. Kvasir: A multi-class image dataset for computer aided gastrointestinal disease detection. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, pages 164–169, 2017.

[66] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[67] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[68] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[69] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[70] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Sparse binary compression: Towards distributed deep learning with minimal communication. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[71] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[72] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *International conference on machine learning*, pages 5907–5915. PMLR, 2019.

[73] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pages 1453–1460. IEEE, 2011.

[74] Nikko Strom. Scalable distributed dnn training using commodity gpu cloud computing. In *Interspeech 2015*, pages 1488–1492, 2015.

[75] Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*, 2018.

[76] Anke Tang, Li Shen, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. Fusionbench: A comprehensive benchmark of deep model fusion. *arXiv preprint arXiv:2406.03280*, 2024.

[77] Qwen Team. Qwen3guard technical report. 2025.

[78] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[79] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[80] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[81] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[82] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, and Furu Wei. Image as a foreign language: BEiT pretraining for vision and vision-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[83] Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

[84] Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

[85] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[86] Haixia Xiao, Feng Zhang, Zhongping Shen, Kun Wu, and Jinglin Zhang. Classification of weather phenomenon from images by using deep convolutional neural network. *Earth and Space Science*, 8(5): e2020EA001604, 2021.

[87] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[88] Prateek Yadav, Leshem Choshen, Colin Raffel, and Mohit Bansal. Compeft: Compression for communicating parameter efficient updates via sparsification and quantization. *arXiv preprint arXiv:2311.13171*, 2023.

[89] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[90] Xiaozhe Yao, Qinghao Hu, and Ana Klimovic. Deltazip: Efficient serving of multiple full-model-tuned llms. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 110–127, 2025.

[91] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024.

[92] Netzer Yuval. Reading digits in natural images with unsupervised feature learning. In *Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[93] Shenghe Zheng and Hongzhi Wang. Free-merging: Fourier transform for model merging with lightweight experts. *arXiv preprint arXiv:2411.16815*, 2024.

 https://doi.org/10.52202/085713-0782

Appendix for UltraDelta

A Non-quantized Variant of Distribution-Aware Compression

While our main Distribution-Aware Compression operates on quantized delta weights, it can also be adapted to a non-quantized variant for scenarios where quantization is not applied or not desired. In this version, we replace the discrete grouping based on quantized values with continuous interval-based grouping over the full-precision delta weights.

Specifically, for a given delta weight $\Delta\theta_t$, we first compute its minimum and maximum values, denoted as $\min(\Delta\theta_t)$ and $\max(\Delta\theta_t)$. We then divide the range $[\min(\Delta\theta_t), \max(\Delta\theta_t)]$ into I equally spaced intervals. The width of each interval is given by:

$$\delta_{\text{interval}} = \frac{\max(\Delta\theta_t) - \min(\Delta\theta_t)}{I} \tag{15}$$

The i-th interval is then defined as:

$$\mathcal{I}i = [\min(\Delta\theta_t) + (i-1) \cdot \delta_{\text{interval}}, \min(\Delta\theta_t) + i \cdot \delta_{\text{interval}})], \quad \text{for } i = 1, \ldots, I \tag{16}$$

To perform group-wise pruning, we define a binary mask for each interval I_i, where pruning decisions are sampled from a Bernoulli distribution with success probability $1 - s_l$, and s_l is the sparsity rate of the layer. Specifically:

$$\boldsymbol{M}_{I_i}^{(j,k)} \sim \begin{cases} \text{Bernoulli}(1 - s_l), & \text{if } {\Delta\theta_t}^{(j,k)} \in I_i \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

The pruned delta weight ${\Delta\theta_t}^*$ is obtained by applying the group-wise Bernoulli masks to the quantized delta weights:

$$\Delta\theta_t^* = \sum_{i=1}^{I} \boldsymbol{M}_{\mathcal{I}_i} \odot \Delta\theta_t \tag{18}$$

B Detailed Derivation

B.1 Derivation of the Variance-Entropy Relationship

We aim to derive the relationship between the variance and the entropy of a layer. First, we define the entropy of a random variable L with probability density function $p(l)$ as:

$$H(L) = -\mathbb{E}[\log p(l)] \tag{19}$$

Assuming L follows a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, its probability density function is:

$$p(l) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(l-\mu)^2}{2\sigma^2}\right) \tag{20}$$

Taking the logarithm of the density function, we obtain:

$$\log p(l) = \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(l-\mu)^2}{2\sigma^2}\right)\right) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{(l-\mu)^2}{2\sigma^2} \tag{21}$$

Substituting this into the definition of entropy, we have:

$$\begin{aligned} H(L) &= -\int_{-\infty}^{\infty} p(l)\left(-\frac{1}{2}\log(2\pi\sigma^2) - \frac{(l-\mu)^2}{2\sigma^2}\right) dl \\ &= \frac{1}{2}\log(2\pi\sigma^2)\int p(l)dl + \frac{1}{2\sigma^2}\int p(l)(l-\mu)^2 dl \end{aligned} \tag{22}$$

Next, we use known properties of the Gaussian distribution:

$$\int p(l)dl = 1 \quad \text{and} \quad \int p(l)(l-\mu)^2 dl = \mathrm{Var}(L) = \sigma^2 \tag{23}$$

Substituting these into the expression for entropy yields:

$$\begin{aligned} H(L) &= \frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2} \\ &= \frac{1}{2}\log(2\pi) + \log(\sigma) + \frac{1}{2} \end{aligned} \tag{24}$$

Therefore, we can conclude that the entropy $H(L)$ increases logarithmically with the standard deviation σ of the distribution. Since the variance is σ^2, entropy is monotonically increasing with variance. This derivation shows that the entropy of a layer is positively correlated with its variance. Since entropy represents the amount of information, layers with larger variance contain more information. As a result, such layers should be assigned a smaller sparsity rate to preserve more information.

B.2 Derivation of the Variance of Activation Error

According to Eq. 13, the activation error ϵ is defined as:

$$\varepsilon = a \odot \left(1 - \frac{\gamma}{1-s}B\right), \quad \text{where } a = \Delta\theta \odot x \tag{25}$$

Here, B is a Bernoulli random variable, where the probability of success is $1-s$. For a Bernoulli distribution, we know that the expectation and second moment are $\mathbb{E}[B] = 1-s$ and $\mathbb{E}[B^2] = 1-s$. We first calculate the expectation of ϵ. Using the linearity of expectation:

$$\begin{aligned} \mathbb{E}[\varepsilon] &= a \odot \mathbb{E}\left[1 - \frac{\gamma}{1-s}B\right] \\ &= a \odot \left(1 - \frac{\gamma}{1-s}\mathbb{E}[B]\right) \\ &= a \odot \left(1 - \frac{\gamma}{1-s}(1-s)\right) \\ &= a \odot (1-\gamma) \end{aligned} \tag{26}$$

Next, we calculate the expectation of ϵ^2:

$$\begin{aligned} \mathbb{E}[\varepsilon^2] &= a^2 \odot \mathbb{E}\left[1 - 2\frac{\gamma}{1-s}B + \left(\frac{\gamma}{1-s}B\right)^2\right] \\ &= a^2 \odot \left[1 - 2\frac{\gamma}{1-s}\mathbb{E}[B] + \left(\frac{\gamma}{1-s}\right)^2 \mathbb{E}[B]\right] \\ &= a^2 \odot \left[1 - 2\frac{\gamma}{1-s}(1-s) + \left(\frac{\gamma}{1-s}\right)^2 (1-s)\right] \\ &= a^2 \odot \left(1 - 2\gamma + \frac{\gamma^2}{1-s}\right) \end{aligned} \tag{27}$$

Now, we can calculate the variance of ϵ. The variance is given by:

$$\begin{aligned} \mathrm{Var}(\varepsilon) &= \mathbb{E}[\varepsilon^2] - (\mathbb{E}[\varepsilon])^2 \\ &= a^2 \odot \left(1 - 2\gamma + \frac{\gamma^2}{1-s}\right) - a^2 \odot (1-\gamma)^2 \\ &= \boldsymbol{a}^2 \odot \left[\left(1 - 2\gamma + \frac{\gamma^2}{1-s}\right) - (1 - 2\gamma + \gamma^2)\right] \\ &= a^2 \odot \left(\frac{\gamma^2 s}{1-s}\right) \end{aligned} \tag{28}$$

https://doi.org/10.52202/085713-0782

From the derived variance expression, we can observe that when the sparsity s is high, the variance will become very large, leading to unstable performance. To mitigate this, we can choose a value for γ in the range $[0, 1]$ to reduce the impact of the original rescaling factor. However, γ cannot be chosen too small. This is because the expectation of the activation error is $a \odot (1 - \gamma)$, and if γ is too small, the expectation of the activation error increases, which may lead to undesirable effects. Therefore, γ is typically set in the range $[0.5, 1.0]$ under high sparsity, which balances both the variance and the expectation of the activation error.

B.3 Detailed Derivation of Compression Ratio

Golomb Coding. Golomb coding [25] is a run-length entropy coding scheme particularly well-suited for a geometric distribution, where it encodes the distances between successive non-zero entries (i.e., run-lengths of zeros). Unlike conventional formats such as Compressed Sparse Row (CSR), which store explicit indices or pointers for non-zero values, run-length coding represents zero runs instead of storing indices, which can yield a more compact representation than CSR when sparsity is high.

Compression Ratio Derivation. In our setting, after pruning, most entries in the delta weight are zeros, and the positions of non-zeros can be modeled as a Bernoulli process with success probability equal to the density $1 - s$, where s is the sparsity. Consequently, the run-lengths of zeros follow a geometric distribution, for which Golomb coding achieves an average code length close to the entropy of the underlying geometric distribution. The entropy of a geometric distribution is:

$$H_{\text{geo}} = -(1 - s) \log_2(1 - s) - s \log_2 s \tag{29}$$

When each non-zero entry can take m discrete values, the maximum entropy contribution of its symbol is $\log_2 m$, which corresponds to the case where all m values are equally likely. In practice, the values of our non-zero entries follow a truncated Gaussian distribution, so the actual entropy is smaller than $\log_2 m$, implying that the achievable compression ratio is higher.

For simplicity of analysis, we approximate this term by $\log_2 m$, which serves as an upper bound, and the per-parameter entropy of the compressed model is:

$$H_{\text{comp}} \approx -\Big[s \log_2 s + (1 - s) \log_2 \tfrac{(1-s)}{m}\Big] \tag{30}$$

By contrast, the original uncompressed model stored in FP16 format requires $H_{\text{orig}} = 16$ bits per parameter. Therefore, the theoretical compression ratio is:

$$\text{CR} = \frac{H_{\text{orig}}}{H_{\text{comp}}} \approx \frac{16}{-\big[s \log_2 s + (1 - s) \log_2 \frac{(1-s)}{m}\big]} \tag{31}$$

Note that the rescaling factor is represented by a single 16-bit value, which is negligible compared to the overall storage and is therefore ignored when reporting the compression ratio.

Detailed Compression Ratios. We also report the compression ratios and corresponding sparsity rates of DARE [91], MP [27], and UltraDelta under each setting. Practical compression ratios with Golomb coding are computed following Eq. 31, using $m = 16$ for DARE and MP (16-bit precision) and $m = 4$ for UltraDelta (4-bit quantization). For completeness, we also include Index-Free compression ratios, which denote the ideal upper bound where only non-zero values are stored. The results are shown in Tab. 8.

C Additional Experimental Settings and Results

C.1 HuggingFace IDs for LLM Checkpoints

We provide the Hugging Face IDs of the LLMs used in our experiments, as shown in Tab. 9. All models are evaluated using the lm-evaluation-harness [23] framework for general language tasks and EvalPlus [52] framework for code generation tasks.

Table 8: Compression ratios, average performance, and corresponding sparsity rates across different models. "Practical" denotes the practical compression ratio using Golomb coding, "Ideal" the ideal upper bound, and "Avg" the average performance.

Method	LLaMA-2-7B (Tab. 1)				LLaMA-2-13B (Tab. 1)			
	Practical	Ideal	Avg	Sparsity	Practical	Ideal	Avg	Sparsity
DARE [91] (Sparsity)	14.7×	20×	38.79	95.0%	23.7×	33.3×	45.63	97.0%
MP [27] (Sparsity)	14.7×	20×	36.46	95.0%	23.7×	33.3×	29.65	97.0%
DARE [91] (Compression)	31.7×	45.5×	33.97	97.8%	50.1×	74.1×	40.23	98.65%
MP [27] (Compression)	31.7×	45.5×	27.02	97.8%	50.1×	74.1×	15.36	98.65%
UltraDelta (Ours)	**32.9**×	**80.0**×	**45.57**	95.0%	**50.9**×	**133.0**×	**52.05**	97.0%

Method	Recent LLMs (Tab. 2)				ViT-B/32 (Tab. 10)			
	Practical	Ideal	Avg	Sparsity	Practical	Ideal	Avg	Sparsity
DARE [91] (Sparsity)	14.7×	20×	49.18	95.0%	23.7×	33.3×	89.7	97.0%
MP [27] (Sparsity)	14.7×	20×	41.68	95.0%	23.7×	33.3×	77.4	97.0%
DARE [91] (Compression)	–	–	–	–	50.1×	74.1×	83.5	98.65%
MP [27] (Compression)	–	–	–	–	50.1×	74.1×	53.1	98.65%
UltraDelta (Ours)	**32.9**×	**80.0**×	**52.05**	95.0%	**50.9**×	**133.0**×	**90.7**	97.0%

Method	ViT-L/14 (Tab. 4)				T5-base (Tab. 3)			
	Practical	Ideal	Avg	Sparsity	Practical	Ideal	Avg	Sparsity
DARE [91] (Sparsity)	66.5×	100.0×	93.6	99.0%	127.6×	200.0×	85.90	99.5%
MP [27] (Sparsity)	66.5×	100.0×	59.7	99.0%	127.6×	200.0×	63.97	99.5%
DARE [91] (Compression)	127.6×	200.0×	89.1	99.5%	220.5×	357.0×	84.30	99.72%
MP [27] (Compression)	127.6×	200.0×	29.5	99.5%	220.5×	357.0×	60.44	99.72%
UltraDelta (Ours)	**132.5**×	**400.0**×	**94.4**	99.0%	**224.6**×	**800.0**×	**86.74**	99.5%

Method	RoBERTa-base (Tab. 11)				BEiT-3 (Tab. 5)			
	Practical	Ideal	Avg	Sparsity	Practical	Ideal	Avg	Sparsity
DARE [91] (Sparsity)	14.7×	20.0×	83.23	95.0%	7.7×	10.0×	70.74	90.0%
MP [27] (Sparsity)	14.7×	20.0×	80.06	95.0%	7.7×	10.0×	16.59	90.0%
DARE [91] (Compression)	31.7×	45.5×	73.12	97.8%	18.1×	25.0×	54.49	96.0%
MP [27] (Compression)	31.7×	45.5×	69.87	97.8%	18.1×	25.0×	16.47	96.0%
UltraDelta (Ours)	**32.9**×	**80.0**×	**84.46**	95.0%	**18.4**×	**40.0**×	**73.95**	90.0%

C.2 Experiments on eight ViT-B/32 models

Results. The results of ViT-B/32 models are shown in Tab. 10. With 4-bit quantization and 97% sparsity, UltraDelta achieves 50.9× compression ratio on ViT-B/32. UltraDelta achieves an average accuracy of 90.7 across 8 diverse vision tasks, nearly matching the performance of individually fine-tuned models (91.0). Compared to all baselines, UltraDelta achieves both higher compression and better accuracy.

C.3 Experiments on eight RoBERTa-base models

Settings. For RoBERTa-base, we report Matthews correlation on CoLA, the average of Pearson and Spearman correlations on STS-B, and accuracy on the other tasks.

Results. The results of RoBERTa-base models are shown in Tab. 11. For RoBERTa, we apply 4-bit quantization and prune 95% of parameters, achieving an overall 32.9× compression ratio. Despite the high compression, UltraDelta achieves an average score of 84.46, closely matching the uncompressed individually fine-tuned model (85.56) and outperforming all baselines by a notable margin.

https://doi.org/10.52202/085713-0782

Table 9: LLM checkpoint IDs.

Models	Pre-trained	Fine-tuned
Model Size 7B (Llama-2 Series)		
Chat	*meta-llama/Llama-2-7b-hf*	*meta-llama/Llama-2-7b-chat-hf*
Code	*codellama/CodeLlama-7b-Python-hf*	*vanillaOVO/WizardCoder-Python-7B-V1.0*
Math	*meta-llama/Llama-2-7b-hf*	*WizardLMTeam/WizardMath-7B-V1.0*
Model Size 13B (Llama-2 Series)		
Chat	*meta-llama/Llama-2-13b-hf*	*meta-llama/Llama-2-13b-chat-hf*
Code	*codellama/CodeLlama-13b-Python-hf*	*WizardLMTeam/WizardCoder-Python-13B-V1.0*
Math	*meta-llama/Llama-2-13b-hf*	*vanillaOVO/WizardMath-13B-V1.0*
Recent Models		
Llama-3.1	*meta-llama/Llama-3.1-8B*	*allenai/Llama-3.1-Tulu-3-8B-SFT*
Qwen2.5	*Qwen/Qwen2.5-7B*	*Qwen/Qwen2.5-7B-Instruct*
Qwen3	*Qwen/Qwen3-8B*	*Qwen/Qwen3Guard-Gen-8B*

Table 10: Performance comparison on ViT-B/32 models across 8 tasks.

Methods	**Ratio**	**SUN397**	**Cars**	**RESISC45**	**EuroSAT**	**SVHN**	**GTSRB**	**MNIST**	**DTD**	**Avg**
Individual	1×	79.2	77.7	96.1	99.8	97.4	98.7	99.7	79.4	91.0
Traditional MTL [8]	8×	73.9	74.4	93.9	98.2	95.8	98.9	99.5	77.9	89.1
BitDelta [53]	16×	78.3	75.9	95.4	99.3	96.4	98.2	99.2	78.4	90.1
Delta-CoMe [64]	16×	78.5	72.3	95.6	99.6	**97.3**	98.6	99.5	78.3	90.0
DARE [91] (Sparsity)	23.7×	76.0	73.3	95.1	**99.7**	97.0	98.5	99.6	78.1	89.7
MP [27] (Sparsity)	23.7×	58.1	44.2	91.2	96.1	91.3	73.5	98.3	66.7	77.4
DARE [91] (Compression)	50.1×	57.2	55.2	91.3	99.4	96.4	98.2	99.6	71.1	83.5
MP [27] (Compression)	50.1×	18.8	16.4	81.8	88.3	70.3	28.5	77.8	42.7	53.1
UltraDelta (Ours)	**50.9×**	**78.6**	**77.4**	**95.7**	99.4	97.0	**98.7**	**99.7**	**79.0**	**90.7**

C.4 Detailed Results of Ablation Study

C.4.1 Ablation on Distribution-Aware Compression

We present the detailed numerical results of the ablation study conducted on DAC. For brevity, we report only the results under 90% sparsity, as summarized in Tab. 12. Following [32, 93], we use ViT-B/16 as the pre-trained backbone model and evaluate its performance across 30 diverse image classification datasets. Specifically, the datasets are: MNIST [44], CIFAR-10 [41], CIFAR-100 [41], Cars [40], Fashion-MNIST [85], EMNIST [16], STL10 [14], GTSRB [73], SHVN [92], Oxford-IIIT Pet [62], Cats and Dogs [18], Dogs [39], Beans [42], Food-101 [7], Fruits-360 [59], Vegetables [1], MangoLeafBD [2], Flowers Recognition [58], Landscape Recognition [19], Weather [86], DTD [13], EuroSAT [29], RESISC45 [12], SUN397 [87], KenyanFood13 [37], Intel Images [5], Garbage Classification [9], Animal-10N [72], CUB-200-2011 [80], and Kvasir-v2 [65]. These datasets are chosen to comprehensively evaluate generalization across various data distributions and classification challenges. Accuracy is used as the evaluation metric for all datasets.

Table 11: Performance comparison on RoBERTa-base models across 8 tasks.

Methods	**Ratio**	**CoLA**	**SST2**	**MRPC**	**STSB**	**QQP**	**MNLI**	**QNLI**	**RTE**	**Avg**
Individual	1×	60.18	94.04	89.22	90.63	91.41	87.20	92.71	79.06	85.56
BitDelta [53]	16×	36.83	93.12	88.73	84.82	**90.00**	**85.57**	91.73	70.40	80.15
Delta-CoMe [64]	16×	57.67	92.48	87.99	90.53	89.17	82.66	**92.57**	76.9	83.74
DARE [91] (Sparsity)	14.7×	59.30	93.92	88.97	90.53	86.55	77.84	92.13	76.53	83.23
MP [27] (Sparsity)	14.7×	57.69	92.43	84.31	88.18	86.26	74.40	86.80	70.40	80.06
DARE [91] (Compression)	31.7×	58.22	93.69	87.50	90.27	53.47	36.48	89.51	75.81	73.12
MP [27] (Compression)	31.7×	51.89	91.63	78.68	86.44	65.77	53.49	65.64	65.43	69.87
UltraDelta (Ours)	**32.9×**	**62.64**	**94.38**	**89.22**	**90.60**	85.79	82.92	92.15	**77.98**	**84.46**

Table 12: Performance of DAC and DARE on ViT-B/16 models across 30 tasks. "(w/ b-bit)" denotes the quantized setting using b-bit precision, while "(w/o q.)" refers to the non-quantized setting.

Methods	Animal-10N	Beans	Cats and Dogs	Cifar-10	Cifar-100	CUB-200-2011	Dogs	DTD	EMNIST	EuroSAT
Individual	92.46	97.70	99.05	97.88	89.85	84.78	89.91	81.1	94.67	99.07
DARE [53] (w/ 4-bit)	92.26	34.87	98.17	33.39	53.37	84.28	89.55	81.08	94.69	99.07
UltraDelta (w/ 4-bit) (Ours)	92.42	97.32	98.98	58.33	72.14	83.98	89.74	80.82	94.51	99.04
DARE [53] (w/ 8-bit)	92.26	34.87	98.17	33.39	53.37	84.28	89.55	81.08	94.69	99.07
UltraDelta (w/ 8-bit) (Ours)	92.68	98.47	98.66	80.06	78.12	84.54	90.18	81.13	94.61	99.00
DARE [53] (w/o q.)	92.28	97.70	98.04	57.90	56.07	84.19	89.91	81.15	94.58	99.15
UltraDelta (w/o q.) (Ours)	92.30	98.08	98.99	96.77	80.83	84.28	90.00	81.08	94.62	99.07
	Fashion	Flowers	KenyanFood13	Food-101	Fruits-360	Garbage	GTSRB	Intel-Images	Kvasir-v2	LandScape
Individual	93.26	98.19	82.58	87.87	99.64	98.58	95.74	94.87	93.91	94.00
DARE [53] (w/ 4-bit)	84.09	98.19	82.70	38.00	99.64	98.61	7.57	93.33	74.81	92.40
UltraDelta (w/ 4-bit) (Ours)	79.58	98.17	82.21	29.14	99.58	98.54	87.16	90.93	56.47	93.80
DARE [53] (w/ 8-bit)	84.09	98.19	82.70	38.00	99.64	98.61	75.70	93.33	74.81	92.40
UltraDelta (w/ 8-bit) (Ours)	83.85	98.19	82.45	47.03	99.59	98.50	91.96	93.93	89.94	73.80
DARE [53] (w/o q.)	73.69	98.17	82.45	43.93	99.64	98.61	62.49	92.13	77.62	93.40
UltraDelta (w/o q.) (Ours)	90.10	98.12	82.82	61.87	99.63	98.50	95.36	94.13	92.22	94.20
	MangoLeafBD	MNIST	Pet	RESISC45	Cars	STL10	SUN397	SVHN	Vegetables	Weather
Individual	100.00	99.22	92.20	99.00	85.29	99.08	87.50	96.22	100.00	98.19
DARE [53] (w/ 4-bit)	98.47	78.59	91.88	98.90	79.78	99.09	85.24	96.22	91.67	97.62
UltraDelta (w/ 4-bit) (Ours)	85.95	98.93	92.56	98.95	79.74	99.21	85.82	96.21	91.30	80.46
DARE [53] (w/ 8-bit)	98.47	78.59	91.88	98.90	79.78	99.09	85.24	96.22	91.67	97.62
UltraDelta (w/ 8-bit) (Ours)	100.00	98.68	92.37	99.10	80.11	99.16	85.74	96.33	99.67	97.51
DARE [53] (w/o q.)	100.00	64.62	92.26	98.79	79.77	99.09	85.55	96.30	99.83	87.03
UltraDelta (w/o q.) (Ours)	100.00	98.97	92.64	99.13	80.55	99.22	85.90	96.32	99.97	97.90

Average Acc	Individual	DARE (w/ 4-bit)	**UltraDelta (w/ 4-bit) (Ours)**	DARE (w/ 8-bit)	**UltraDelta (w/ 8-bit) (Ours)**	DARE (w/o q.)	**UltraDelta (w/o q.) (Ours)**
Acc	94.06	81.58	**86.40**	83.85	**90.18**	85.88	**92.45**

C.4.2 Ablation on Hyper-Parameters

We present the detailed numerical results of our hyper-parameter ablation studies in Tab. 13. The results complement the analysis provided in the main text and offer full visibility into the performance under different settings.

Table 13: Performance of different hyperparameter settings applied to ViT-B/32.

Method	**SUN397**	**Cars**	**RESISC45**	**EuroSAT**	**SVHN**	**GTSRB**	**MNIST**	**DTD**	**Avg.**
Distribution-Aware Compression (quantization bit)									
2-bit	73.4	70.1	94.4	99.1	96.9	98.2	99.7	74.3	88.3
4-bit	78.6	77.4	95.7	99.4	97.0	98.7	99.7	79.0	90.7
6-bit	78.7	77.5	95.7	99.5	97.1	98.8	99.7	78.6	90.7
8-bit	78.8	77.7	95.7	99.7	97.0	98.7	99.7	79.0	90.8
Variance-Based Mixed Sparsity Allocation (sparsity step size)									
Step = 0.000	77.3	75.1	95.5	99.7	97.1	98.8	99.6	77.6	90.1
Step = 0.005	77.9	75.7	95.4	99.7	97.0	98.8	99.7	78.9	90.4
Step = 0.010	78.4	76.7	95.6	99.4	97.0	98.9	99.6	78.9	90.6
Step = 0.020	78.6	77.4	95.7	99.4	97.0	98.7	99.7	79.0	90.7
Step = 0.025	78.3	76.4	95.1	98.1	96.6	98.6	99.7	79.2	90.3
Step = 0.030	78.5	75.3	94.5	96.5	96.7	98.2	99.7	78.7	89.8

D Discussion

D.1 Regularization Effect

We observe in experiments that UltraDelta sometimes even outperforms fine-tuned models, especially in the LLM setting. This can be attributed to overfitting or underfitting during fine-tuning.

To verify this, we conducted controlled experiments on two datasets, SUN397 [87] and Cars [40], using a pretrained ViT-B/32 model. By varying the number of fine-tuning steps, we controlled the degree of model fitting. The results are shown in Tab. 14, and we find that underfitted models (early training steps, higher loss) consistently benefit from UltraDelta, while well-trained models (later steps, lower loss) show some or no improvement. These results indicate that UltraDelta mainly benefits underfitted models, while well-trained models already capture most task-specific information and thus leave little room for further improvement.

https://doi.org/10.52202/085713-0782

This also explains why improvements are more common in LLMs, which are typically evaluated in zero-shot or few-shot settings. In such scenarios, these models are not explicitly fine-tuned on the target tasks, which often leads to weaker generalization. As a result, they can be regarded as underfitted to the evaluation benchmarks. In this context, UltraDelta provides a regularization effect, enabling underfitted models to benefit more from compression.

Table 14: Controlled experiments on ViT-B/32 with varying training steps. "UltraDelta Accuracy" denotes the test accuracy after applying UltraDelta.

Training Steps	**1000**	**2000**	**3000**	**4000**	**5000**	**6000**	**7000**	**8000**	**9000**	**10000**
Cars [40]										
Training Loss	0.1404	0.0834	0.0206	0.0069	0.0059	0.0059	0.0034	0.0077	0.0048	0.0036
Test Accuracy	72.86	68.90	75.22	**78.78**	**77.74**	**78.37**	**78.01**	**75.21**	**77.60**	**77.34**
UltraDelta Accuracy	**74.02**	**73.27**	**75.82**	75.56	75.62	75.66	76.13	72.80	74.74	74.06
SUN397 [87]										
Training Loss	0.5475	0.1432	0.0426	0.1125	0.0526	0.0040	0.0018	0.0009	0.0006	0.0004
Test Accuracy	73.63	75.94	75.40	74.63	74.07	**76.94**	76.87	77.03	77.06	77.07
UltraDelta Accuracy	**76.02**	**76.92**	**77.21**	**76.20**	**76.19**	76.66	**76.94**	**77.11**	**77.19**	**77.31**

D.2 Compression Ratio Differences Across Models

We observe that different models can be compressed to different extents without losing performance. This variation mainly comes from the amount of redundancy in their delta weights. For instance, T5-base has relatively high redundancy, making it possible to compress it up to 224$\times$ while maintaining or even improving performance. In contrast, models like BEiT-3 have delta weights with larger magnitudes and lower redundancy, leaving less room for aggressive compression. This suggests that the achievable compression ratio depends not only on the compression method itself, but also on the model architecture, its training process, and the nature of the fine-tuned tasks.

D.3 Limitations and Future Work

Heuristic Rescaling Factor. One limitation of our method lies in the heuristic nature of the additional rescaling factor γ. We set γ according to the trace norm of each delta weight, which serves as a lightweight proxy for activation instability under the data-free setting. While this simple strategy helps mitigate instability at high sparsity, it remains a heuristic and may not be optimal. Exploring more principled or adaptive approaches for determining γ under data-free constraints remains an important direction for future work.

Deployment Efficiency. Another limitation is that our method is not designed to directly optimize deployment efficiency, such as inference acceleration. Achieving real speedup typically requires specialized GPU kernels or structured sparsity to better exploit hardware parallelism. Future work may combine our approach with optimized kernels or structured pruning to realize both strong compression–performance trade-offs and practical deployment gains.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly summarize our main contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations and potential future work in the Discussion section and the Appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

https://doi.org/10.52202/085713-0782

Answer: [Yes]

Justification: For all theoretical results in this paper, we provide detailed derivations.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide detailed descriptions of the experimental settings and evaluation metrics in Experiments section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code has been released at https://github.com/xiaohuiwang000/UltraDelta.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All experimental settings, including model types, datasets, hyper-parameters, are reported in Experiments section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper does not report error bars or other statistical significance measures.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

https://doi.org/10.52202/085713-0782

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the type of GPU in the paper. All experiments are conducted on a single NVIDIA A800.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We ensure compliance with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This paper presents a theoretical/methodological contribution without application to real-world scenarios. Thus, it may not have societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work does not involve pretrained generative models or other high-risk assets. Therefore, safeguards are not necessary.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets are properly cited in the references section.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

https://doi.org/10.52202/085713-0782

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not introduce any new assets such as datasets, models, or code.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our research does not involve crowdsourcing or experiments involving human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our study does not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/085713-0782