BitDP: Ultra-low-bit Communication for Data Parallelism in LLM Training

Xiaozhe Ren, Qiong Luo

Hong Kong University of Science and Technology
xrenac@connect.ust.hk, luo@cse.ust.hk

Abstract

Training large language models (LLMs) with billions of parameters on trillion-token datasets requires distributed data parallelism at increasingly large scales, where gradient synchronization becomes a communication bottleneck, especially in bandwidth-constrained environments. Although gradient quantization presents a promising solution, it faces two key challenges: maintaining training stability and accuracy for transformer architectures and adapting to modern distributed communication systems. In this paper, we propose BitDP, an ultra-low-bit gradient quantization system that reduces communication costs by up to 32× while preserving model accuracy with less than 1% performance degradation. Our approach achieves numerical stability for large transformer models and seamlessly integrates with existing infrastructures. We evaluate BitDP's effectiveness across various LLM sizes, architectures and optimizers. The results demonstrate significant training efficiency improvements while maintaining convergence quality, establishing BitDP as a scalable and reliable solution for real-world LLM training at industrial scales.

Introduction

Large Language Models (LLM) such as GPT-3 (Brown et al. 2020), GPT-4 (Achiam et al. 2023), Gemini (Team et al. 2023), and DeepSeek-R1 (Guo et al. 2025) have been making rapid advances. These models, built on scalable transformer architectures, have achieved remarkable performance in tasks such as natural language processing and multimodal generation. However, their immense scale introduces significant technical challenges, particularly in distributed training, where communication is one of the major performance bottlenecks (Rajbhandari et al. 2020; Narayanan et al. 2021).

A major challenge in distributed LLM training lies in the exchange of gradients during backpropagation. As model sizes grow, the volume of gradient data increases correspondingly, leading to heavy network bandwidth demands and sub-optimal utilization of computational resources. This problem is particularly pronounced at scale, where communication overhead often dominates computation time.

To address this issue, gradient compression techniques such as quantization (Wen et al. 2017; Alistarh et al. 2017; Wang et al. 2024; Jia et al. 2024) and sparsification (Lin et al. 2018; Shi et al. 2021) have been proposed. However, these methods have not been widely applied to LLMs due to limited compression rates or concerns about training accuracy. Achieving ultra-low-bit compression, such as 1-bit or 2-bit gradients, while maintaining training accuracy for LLMs remains a significant challenge. Moreover, current communication libraries such as the NVIDIA Collective Communications Library (NVIDIA Corporation 2020) lack native support for such low-bit operations, further complicating their practical deployment.

In this paper, we introduce **BitDP**, a system for ultra-low-bit gradient communication in LLM training (Figure 1).

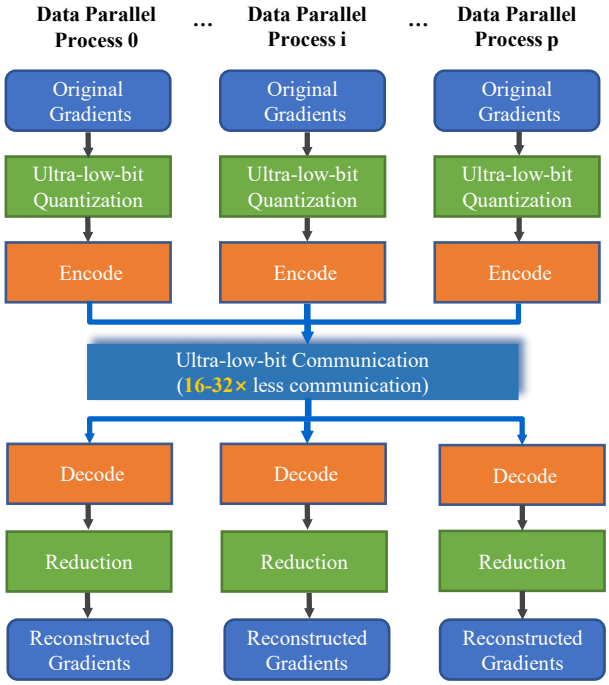

Figure 1: Workflow of BitDP ultra-low-bit data parallel training. The proposed system reduces communication cost by up to 32× while preserving large language model training accuracy across distributed nodes.

BitDP achieves up to 32× reduction in communication overhead by employing novel 1-bit or 2-bit gradient quantization, coupled with a specialized ultra-low-bit gradient communication algorithm compatible with existing communication interfaces. The main contributions of this paper are as follows:

- We propose BitDP, a comprehensive system for efficient LLM training that incorporates ultra-low-bit (1-bit and 2-bit) gradient quantization techniques, reducing communication overhead by up to 32× while maintaining comparable model performance.
- We analyze the gradient distributions of LLMs and propose an adaptive gradient compression algorithm that derives optimal analytical solutions for 1-bit and 2-bit quantization based on the identified distribution patterns, eliminating the need for manual hyperparameter tuning.
- We develop a novel ultra-low-bit gradient communication system as a core component of BitDP to enable ultra-low-bit communication, leveraging existing communication frameworks for practical deployment while preserving system efficiency.
- We validate the entire BitDP system with extensive experiments on various LLMs and optimizers, demonstrating its effectiveness and scalability across different model sizes and architectures without compromising convergence behavior, while achieving significant throughput improvements across diverse network bandwidth scenarios.

Related Work

In this section, we review the existing literature on gradient quantization and gradient sparsification, which are closely related to our work.

Gradient Quantization Gradient quantization techniques reduce communication overhead by encoding gradients using fewer bits. Early approaches like QSGD (Alistarh et al. 2017) and TernGrad (Wen et al. 2017) showed promise for vision tasks but were primarily designed for parameter server architectures, not the AllReduce-based systems for modern LLM training. Additionally, these methods lacked validation on transformers, where numerical stability presents unique challenges (Chowdhery et al. 2023).

Recent LLM-specific quantization methods such as Zero++ (Wang et al. 2024) and SDP4bit (Jia et al. 2024) achieve results comparable to full-precision training, but only reduce to 4 bits per gradient element. In contrast, BitDP pushes boundaries to ultra-low 1-2 bit communication while maintaining training stability and integrating with modern collective communication interfaces, representing a substantial advancement over existing approaches.

Gradient Sparsification Gradient sparsification reduces communication overhead by transmitting only the most significant gradient elements (Lin et al. 2018; Shi et al. 2021). These methods prioritize critical gradient information to maintain convergence while decreasing bandwidth requirements. However, despite these benefits, the selection and tracking of significant gradient elements often requires complex thresholding mechanisms that add computational overhead. Furthermore, these techniques lack validation on large-scale LLM training and integration with modern distributed training infrastructures, limiting their practical applicability for LLM training scenarios.

Parameter Name	Gradient Volume
`Word Embeddings`	vd
`Position Embeddings`	sd
`Layernorm Weight`	$(2n+1)d$
`Layernorm Bias`	$(2n+1)d$
`ATT QKV weight`	$3nd^2$
`ATT QKV Bias`	$3nd$
`ATT Linear weight`	nd^2
`ATT Linear Bias`	nd
`MLP d_4d Weight`	$4nd^2$
`MLP d_4d Bias`	$4nd$
`MLP 4d_d Weight`	$4nd^2$
`MLP 4d_d Bias`	nd

Table 1: Gradient communication volume per data parallel in standard GPT model training. n is the number of layers, d is the hidden dimension, v is the vocabulary size, and s is the sequence length.

Method

An effective ultra-low-bit gradient compression technique should simultaneously address two key challenges: minimizing accuracy degradation to maintain model performance, and enabling efficient ultra-low-bit gradient synchronization to reduce communication overhead.

In this section, we introduce the two key components of our approach to address the aforementioned challenges. First, we present our Fine-Grained Gradient Quantization Algorithm, which enables precise control of gradient data at ultra-low-bit levels, focusing compression on gradients of linear layer weights that are both communication-dominant and quantization-tolerant. Second, we describe our ultra-low-bit gradient communication method, which leverages advanced collective communication primitives and hardware capabilities to significantly reduce gradient synchronization overhead. Together, these components achieve a reduction factor of 16 to 32 times in communication volume without compromising training accuracy.

Fine-grained ultra-low-bit Gradient Quantization

Selective Compression for Communication-Dominant Linear Layers The parameter gradient volumes for various components of a standard GPT model are summarized in Table 1. In large language model (LLM) training, most of the gradient communication volume comes from gradients of attention QKV weights, attention linear weights, and MLP weights, which scale quadratically with the hidden dimension. In contrast, smaller components such as biases and layer normalization contribute minimally. For instance, in GPT-3 (Brown et al. 2020), the four largest dense weights account for 99.6% of the total gradient volume. In LLaMA (Touvron et al. 2023), this proportion is even

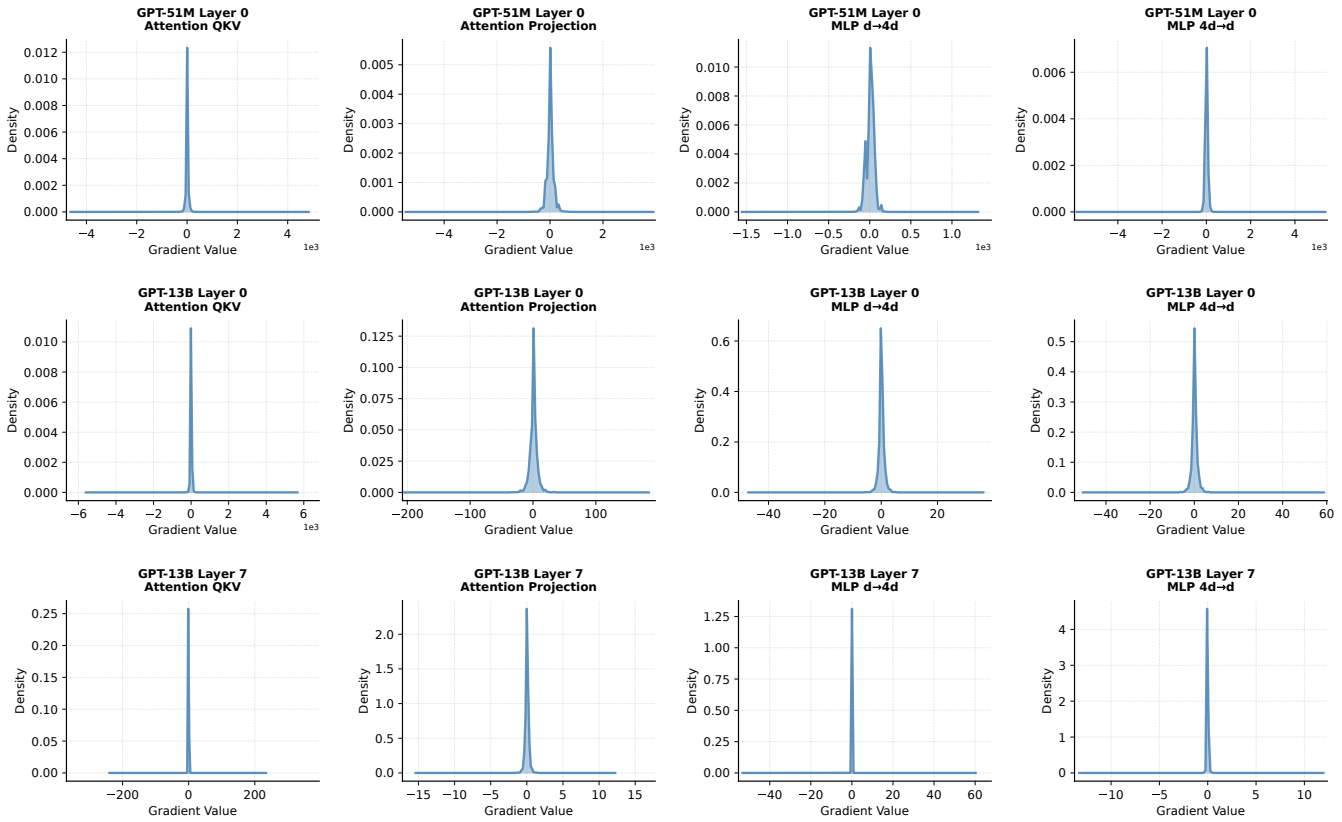

Figure 2: Overlaid gradient density distributions of GPT-51M (51 million parameters) and GPT-13B (13 billion parameters) models across 20,000 training steps. Each subplot shows the aggregated gradient density distribution sampled every 50 steps during training for different Transformer components. The first row corresponds to GPT-51M at Layer 0, the second row represents GPT-13B at Layer 0, and the third row shows GPT-13B at Layer 7. From left to right, the columns display Attention QKV, Attention Projection, MLP $d \rightarrow 4d$, and MLP $4d \rightarrow d$ components, respectively.

higher due to optimizations like rotary position embedding (Su et al. 2024) and RMSNorm (Zhang and Sennrich 2019).

Our empirical analysis reveals that different gradient types exhibit distinct quantization tolerance (detailed in the experiment section). While linear layer gradients demonstrate high quantization tolerance, aggressive quantization of bias terms and embedding gradients leads to training instability and convergence degradation. Although layer normalization gradients also exhibit quantization tolerance similar to linear layers, their communication overhead is negligible compared to massive linear transformations, making them less critical for compression optimization.

In our proposed gradient quantization strategy, we quantize the gradients of all linear weights to ultra-low-bit precision, such as 2-bit or 1-bit, while maintaining full precision for all other components' gradients including bias, embedding weights, and normalization layers. This selective approach targets the communication-dominant components while ensuring training stability.

Gradient Distribution Analysis of LLM Linear Layers In this section, we analyze the gradient distributions of linear layers in LLMs across different model sizes and layers. Specifically, we focus on two representative models: GPT-51M and GPT-13B, which differ significantly in model scale. GPT-51M, a small-scale model, consists of 8 Transformer layers with a hidden dimension of 512. In contrast, GPT-13B, a large-scale model, contains 40 Transformer layers with a hidden dimension of 5120. The analysis spans 20,000 training steps, with gradients sampled every 50 steps to capture the overall density distribution. Figure 2 visualizes the overlaid density distributions of weight gradients for all linear sublayers within a single Transformer layer. This figure provides insight into how gradient distributions evolve across models of different scales and depths.

From the analysis, we observe that the gradients exhibit a distinct **triangular distribution**, symmetrically centered around zero, across both small and large scale models. This distribution reflects stable gradient propagation throughout the training, as the gradients remain balanced without becoming excessively concentrated or dispersed. In particular, for larger models, such as GPT-13B, we observe that they align more closely with our assumed gradient distribution. Such alignment makes them better suited for applying our quantization algorithms, further reinforcing the scalability of our approach.

Channel-adaptive Ultra-low-bit Gradient Quantization A generalized gradient quantization algorithm f_{quant} maps the full-precision gradient $\mathbf{g}$ to a scaling factor $\mathbf{s}$ and a low-precision quantized gradient $\mathbf{q}$:

$$(\mathbf{s}, \mathbf{q}) = f_{\text{quant}}(\mathbf{g}) \tag{1}$$

This process preserves essential gradient information while reducing precision for efficient communication.

Traditional methods use a single scaling factor for the entire gradient tensor such as TernGrad (Wen et al. 2017) may lose important gradient magnitude information. Some recent works such as Zero++ (Wang et al. 2024) and SDP4bit (Jia et al. 2024) adopt fixed grouped quantization strategies that quantize gradients to 4 bits within pre-defined group sizes, but these methods require tuning the group size on a case-by-case basis.

To address this limitation, we propose channel-adaptive ultra-low-bit quantization, which applies channel-wise scaling factors instead of a single tensor-wide factor or fixed group sizes. For a full precision gradient tensor $\mathbf{g} \in \mathbb{R}^{m \times n}$, the goal is to approximate it as $\mathbf{g} \approx \mathbf{s} \odot \mathbf{q}$, where $\mathbf{s} \in \mathbb{R}^m$ is a channel-wise scaling vector and $\mathbf{q} \in \mathbb{I}^{m \times n}$ is a quantized tensor, $\odot$ denotes the Hadamard product. The elements of $\mathbf{q}$ are constrained to $\{-1, 1\}$ for 1-bit quantization and $\{-1, 0, 1\}$ for 2-bit quantization.

To minimize the quantization error, we solve the following optimization problem:

$$\min_{\mathbf{s},\mathbf{q}} L(\mathbf{s}, \mathbf{q}) = \|\mathbf{g} - \mathbf{s} \odot \mathbf{q}\|_2^2 \tag{2}$$

This objective reduces the squared Euclidean distance between the original gradient $\mathbf{g}$ with corresponding quantized approximation $\mathbf{q}$.

This optimization can be solved independently for each channel i. The optimal scaling factor s_i^* is derived by minimizing L with respect to s_i:

$$s_i^* = \frac{1}{|\Omega_{\alpha_i}|} \sum_{j \in \Omega_{\alpha_i}} |g_{i,j}| \tag{3}$$

Here, Ω_{α_i} is the set of indices where $|g_{i,j}| > 0$ for channel i, and $|\Omega_{\alpha_i}|$ represents the number of non-zero elements.

For 1-bit quantization, the quantized tensor $\mathbf{q}$ is defined as:

$$q_{i,j} = \begin{cases} 1, & \text{if } g_{i,j} \geq 0 \\ -1, & \text{if } g_{i,j} < 0 \end{cases} \tag{4}$$

The corresponding scaling factor simplifies to the mean of absolute values:

$$s_i^* = \frac{1}{n} \sum_{j=1}^{n} |g_{i,j}| \tag{5}$$

For the 2-bit variant, the quantization function $\mathbf{q}$ introduces sparsity by mapping small gradient values to zero. It is defined as:

$$q_{i,j} = \begin{cases} 1, & \text{if } g_{i,j} \geq \alpha_i \\ 0, & \text{if } -\alpha_i \leq g_{i,j} < \alpha_i \\ -1, & \text{if } g_{i,j} < -\alpha_i \end{cases} \tag{6}$$

Algorithm 1: BitDP: Ultra-low-bit Data Parallel Training

Require: Gradient tensors of all N modules in model $g = \{g^0, ..., g^N\}$, quantization tensor list $List$, bit mode $b \in \{1, 2\}$, threshold factor $\beta = \frac{3}{4}$, data parallel size p

for $name, g^k$ in g **do**
  if $name$ in $List$ **then**
    for channel i in g^k **do**
      if $b == 1$ **then**
        $q_i^k = Q_{1bit}(g_i^k)$
        $s_i^k = \frac{\sum_j |g_{i,j}|}{n}$
      else
        $\alpha_i = \beta \mathbb{E}_j[|g_{i,j}|]$
        $q_i^k = Q_{2bit}(g_i^k, \alpha_i)$
        $s_i^k = \frac{\sum_j |g_{i,j}|}{\sum_j |q_{i,j}|}$
      end if
    end for
    $q_{ag}^k = UltraLowBit\text{-}Comm(q^k)$
    {AllGather-based Communication}
    $s_{ar}^k = AllReduce(s^k)$
    $q_{ar}^k = DecodeAndReduce(q_{ag}^k)$
    $g_{ar}^k = s_{ar}^k \odot q_{ar}^k / p$
  else
    $g_{ar}^k = AllReduce(g^k)/p$
  end if
end for

where α_i is a channel-specific threshold. This threshold determines the sparsity level, and when $\alpha_i = 0$, Equation (6) reduces to the 1-bit quantization function in Equation (4).

The optimal scaling factor $\mathbf{s}$ is obtained by substituting Equation (3) into Equation (2). The threshold α_i can be optimized by maximizing the following objective:

$$\max_{\alpha_i} \frac{1}{|\Omega_{\alpha_i}|} \left(\sum_{j \in \Omega_{\alpha_i}} |g_{i,j}| \right)^2 \tag{7}$$

where Ω_{α_i} denotes the set of indices where $|g_{i,j}| > \alpha_i$.

Since the magnitude of gradient g approximately follows a symmetric triangular distribution centered at zero, where its probability density function (PDF) is:

$$f(g_i) = \frac{h_i - |g_i|}{h_i^2}, \qquad g_i \in [-h_i, h_i], \tag{8}$$

the optimal solution of α_i can be derived from Equation (7) as:

$$\alpha_i^* = \frac{h_i}{4} = \frac{3}{4} \mathbb{E}_j[|g_{i,j}|] \tag{9}$$

where $h_i = \max_j(|g_{i,j}|)$ is the maximum gradient magnitude in channel i, and $\mathbb{E}_j[|g_{i,j}|]$ is the mean absolute gradient value. We use $\mathbb{E}_j[|g_{i,j}|]$ in practice to reduce sensitivity to outliers.

Based on the ultra-low-bit gradient quantization scheme described above, we further propose a novel ultra-low-bit communication system that transforms our theoretical communication savings into practical reality. While the detailed

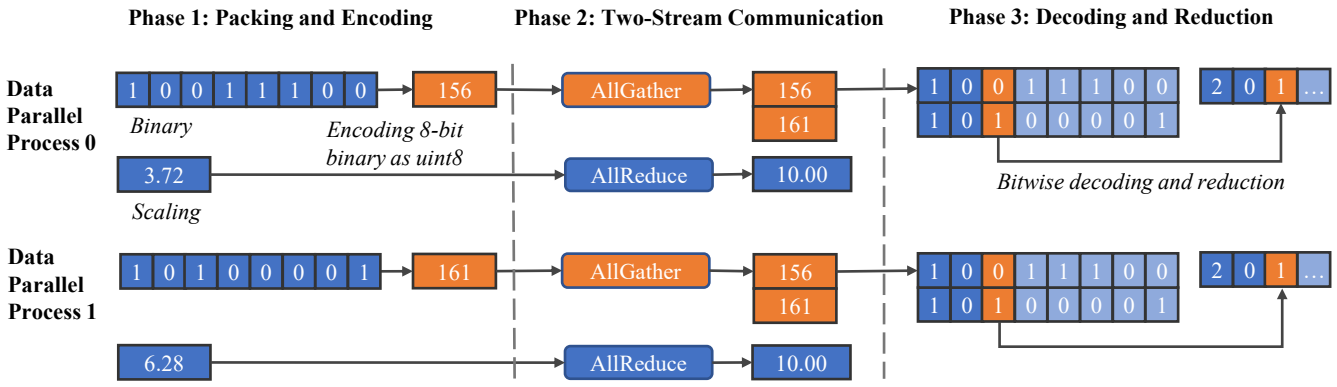

Figure 3: Illustration of ultra-low-bit communication system. In Phase 1, quantized gradients (shown here with 1-bit quantization for clarity) along with their corresponding scaling factors are processed, with every 8 binary bits packed into a single UINT8 value to optimize communication efficiency. Phase 2 performs parallel communication where scaling vectors are synchronized using full-precision AllReduce operations while the compressed gradient matrices use UINT8-based AllGather operations. Phase 3 decodes and reduces gradients bit by bit, avoiding full decompression to save memory.

communication mechanism will be elaborated in the next subsection, we present the complete BitDP algorithm flow in Algorithm 1, which integrates both the fine-grained ultra-low-bit gradient quantization and the ultra-low-bit communication procedure. Specifically, the quantization tensor list includes all linear layers in the model, excluding biases, embedding layers, and normalization layers.

Ultra-low-bit Gradient Communication

In distributed LLM training, gradient synchronization typically uses FP32 AllReduce operations to prevent numerical overflow. However, existing frameworks like NCCL lack native support for ultra-low precision gradients (1-bit or 2-bit), creating inefficiencies in communication for such formats.

Quantized gradients consist of a channel-wise FP32 scaling vector and ultra-low precision (1-bit or 2-bit) tensors. While FP32 vectors can be synchronized using standard AllReduce, synchronizing quantized tensors is more challenging. Representing low-precision values as INT8 formats is inefficient, wasting 75% of storage space for 2-bit gradients and risking overflow when the number of workers exceeds 127.

To overcome these challenges, we propose specialized ultra-low-bit communication algorithms for 1-bit and 2-bit gradients. Key techniques include bit-level gradient packing, efficient AllGather-based communication, two-stream synchronization and reduction. These methods fully exploit the bandwidth savings of ultra-low-bit quantization while enabling efficient gradient communication. The overall workflow of the proposed methods is illustrated in Figure 3.

- **Bit-level Packing and Encoding** For 1-bit quantization, gradient values are first mapped from $\{-1, 1\}$ to $\{0, 1\}$ for efficient storage. These binary values are grouped into buckets of 8 and packed into a single UINT8 byte using bit-wise encoding. For 2-bit quantization, we separate the original gradient into two matrices: one retaining -1 values and another retaining 1 values, setting all other elements to 0. The -1 matrix is mapped from $\{-1, 0\}$ to $\{1, 0\}$. Both matrices are then packed into UINT8 integers using the same bit-wise strategy as in the 1-bit case.
- **AllGather Based Gradient Communication** To prevent overflow, we divide traditional gradient AllReduce into two steps: AllGather followed by local reduction. During AllGather, each data-parallel rank exchanges encoded gradients with all other ranks using efficient UINT8 collective communication primitives. Since no arithmetic operations occur in this phase, overflow is avoided and all units receive a complete copy of the encoded gradients for subsequent reduction.
- **Two-Stream Communication and Reduction** We synchronize the quantized gradients and scaling vectors in parallel. Quantized gradients are first aggregated using the AllGather operation, then decoded and reduced bit by bit to avoid storing fully decompressed gradients. Scaling vectors are synchronized via AllReduce. With the reduced quantized gradient $q_{\mathtt{ar}}$ and reduced scaling vector $s_{\mathtt{ar}}$, the final gradient is computed as:

$$g_{\mathtt{ar}} = \frac{1}{p} s_{\mathtt{ar}} \odot q_{\mathtt{ar}} \tag{10}$$

where p is the data-parallel group size.

Experiments

In this section, we present a comprehensive experimental evaluation of our BitDP algorithm and system. First, we evaluate the training performance of our method through ablation studies and comparisons with existing approaches such as QSGD, TernGrad, Zero++, and SDP4bit. Second, we validate the scalability and generalization of our algorithm by evaluating it across varying model sizes, transformer variants and optimizer. Third, we evaluate the system

Method	Valid. PPL	Compress. Ratio
Baseline	22.17	1
BitDP-2bit	22.74	16
BitDP-1bit	23.40	32
BitDP-2bit w/LN	23.22	16
BitDP-2bit Full	Diverge	16
BitDP-2bit w/Embd	Diverge	16
BitDP-2bit w/bias	Diverge	16
QSGD-2bit	43.20	16
TernGrad	36.45	16
Zero++	22.23	8
SDP4bit	22.35	8

Table 2: Performance evaluation of GPT-350M with different gradient quantization methods on validation perplexity and compression ratio. Baseline uses FP32 gradients. BitDP-2bit and BitDP-1bit represent our proposed fine-grained quantization methods. The middle section shows ablation study results, and the bottom section compares with other gradient quantization approaches including QSGD, TernGrad, Zero++, and SDP4bit.

performance across diverse network bandwidth scenarios. All experiments are conducted using the PyTorch (Paszke et al. 2019) and Megatron-LM (Shoeybi et al. 2019) frameworks on NVIDIA V100 GPUs.

Training Performance Evaluation

Experiments Setting We validated the training accuracy of BitDP using a GPT-3 (Brown et al. 2020) model with 350 million parameters. The model based on the standard Transformer decoder architecture, was trained on a mixed dataset of BooksCorpus (Zhu et al. 2015), English Wikipedia, and OpenWebText (Gokaslan and Cohen 2019) which are aligned with BERT (Devlin et al. 2019) and GPT-2 (Radford et al. 2019), distributed across 8 GPUs. All experiments were conducted with a global batch size of $65,536$ tokens for 50,000 steps. Hyperparameters were aligned with the original GPT-3 configuration and kept consistent across all evaluations. The AdamW optimizer was used with a learning rate of $3e-4$ and a weight decay of 0.01.

Ablations on Layer Selection Table 2 summarizes the training performance of our method through ablation studies and comparisons with existing methods. The baseline model using FP32 gradients achieves a validation perplexity of 22.17. Our proposed BitDP-2bit method maintains competitive performance with a validation perplexity of 22.74 while achieving a 16× compression ratio, and BitDP-1bit delivers 32× compression with 23.40 perplexity. These results demonstrate that our fine-grained gradient quantization achieves substantial compression with minimal impact on model performance.

The ablation study reveals the importance of our selective quantization strategy. Full 2-bit quantization across all modules, as well as including embedding or bias gradients, results in training divergence. While incorporating layer normalization gradients achieves comparable performance, it provides minimal compression benefits due to the small gradient volume. Therefore, BitDP-2bit remains the optimal approach balancing performance and efficiency.

Model	Method	Optimizer	Valid. PPL
GPT-2.7B	Baseline	Adam	18.60
GPT-2.7B	BitDP-2bit	Adam	18.92
GPT-2.7B	BitDP-1bit	Adam	21.25
GPT-350M	Baseline	Came	22.41
GPT-350M	BitDP-2bit	Came	23.07
GPT-350M	BitDP-1bit	Came	23.69

Table 3: Validation perplexity comparison of BitDP methods across different model sizes and optimizers

Comparison with Existing Methods The comparison with existing gradient quantization methods, presented in the bottom section of Table 2, demonstrates the superiority of our approach across different compression regimes. Compared to early 2-bit quantization methods, our BitDP-2bit achieves significantly better accuracy, with a validation perplexity of 22.74 versus 43.2 for QSGD-2bit and 36.45 for TernGrad. When compared to recent quantization methods like Zero++ (22.23 perplexity) and SDP4bit (22.35 perplexity), our BitDP maintains comparable accuracy while achieving up to 32× compression ratio compared to the 8× compression of existing 4-bit methods.

This substantial improvement comes from fundamental differences in quantization strategies. Stochastic quantization methods like QSGD and TernGrad fail to fully exploit the natural distribution characteristics of gradients, whereas our approach is specifically designed to leverage these distributions for more effective quantization at extremely low bit widths. In contrast to fixed-grouping quantization methods like Zero++ and SDP4bit, our adaptive grouping strategy leverages channel-wise information, enabling greater flexibility and ease of use. From a Pareto frontier perspective, our method represents the current optimal trade-off between accuracy and compression efficiency.

Scalability and Generalization Evaluation

Experiments on GPT-2.7B To evaluate the scalability of our algorithms, we tested the pre-training performance of a GPT-3 model with 2.7 billion parameters. This model shares the same architecture as the 350M model but features an expanded dimension of 2560, 32 layers, and 32 attention heads, consistent with the GPT-3 2.7B specification. The experiments were conducted on 32 GPUs distributed across 4 servers, using the same dataset as the 350M model. As shown in Table 3, the BitDP-2bit model achieves performance close to the full-precision model of the baseline, with only 0.32 increase in validation perplexity. Notably, this gap is smaller than that observed in the 350M model, where the difference is 0.57 in perplexity. These results highlight the scalability of the BitDP-2bit approach.

Experiments on CAME optimizer To further validate the effectiveness of BitDP across different optimizers, we conducted experiments with the CAME optimizer (Luo et al.

Method	ARC-E	BoolQ	H-Swag	PIQA	COPA	REC	Avg
Baseline	47.3	56.7	31.3	65.2	69.0	73.0	57.1
BitDP-2bit	47.6	57.0	30.6	63.8	72.0	71.6	57.1
BitDP-1bit	47.3	55.9	30.4	63.9	72.0	71.0	56.8

Table 4: Downstream tasks performance evaluation of LLaMA-8B on various benchmarks (accuracy in %). Baseline uses FP32 gradient, while BitDP-2bit and BitDP-1bit represent our proposed fine-grained 2-bit and 1-bit fine-grained quantization methods respectively. Benchmarks include ARC-Easy (ARC-E) (Clark et al. 2018), BoolQ (Clark et al. 2019), HellaSwag (H-Swag) (Zellers et al. 2019), Physical Interaction QA (PIQA) (Bisk et al. 2020), Choice of Plausible Alternatives (COPA) (Roemmele, Bejan, and Gordon 2011), and ReCORD (REC) (Zhang et al. 2018).

2023) on the 350M parameter GPT model. Table 3 presents the validation perplexity results under different gradient quantization settings. The results demonstrate that BitDP maintains excellent performance when using the CAME optimizer. With 2-bit quantization (BitDP-2bit), CAME achieves a validation perplexity of 23.07, representing only a modest 0.66-point increase from the full-precision baseline. Even under aggressive 1-bit quantization, CAME maintains relatively stable performance with a perplexity of 23.69, resulting in a performance gap of only 1.28 points compared to the baseline.

Experiments on LLaMA-8B To further evaluate the generalization of our BitDP method across different architectures and larger model, we conducted experiments on LLaMA (Touvron et al. 2023), an 8B parameter model with a distinctly different architecture from GPT. The model features 32 layers, 4096 hidden dimensions. We trained each model configuration for 50,000 steps with 65,536 tokens per batch on 64 GPUs distributed across 8 servers. The datasets remained consistent with our previous experiments.

Evaluation on downstream tasks yields promising results. As shown in Table 4, our 2-bit quantization maintains the baseline's 57.1% average performance across all tasks, while 1-bit quantization achieves 56.8% average performance, with only a 0.5% degradation compared to full precision. This remarkable result can be attributed to the bias-free architecture. LLaMA's elimination of bias terms in linear layers removes this quantization-sensitive component, significantly narrowing the performance gap between ultra-low-bit and full-precision training. This finding demonstrates that BitDP is particularly effective for advanced architectures, showing improved quantization efficiency as model size increases.

System Performance Evaluation

We evaluate the end-to-end training throughput of BitDP on a GPT-13B model using 64 GPUs with data parallelism degree of 8 and tensor parallelism degree of 8, with 10 measurements collected for each experiment to ensure statistical reliability as indicated by the reported standard deviations. Table 5 reveals that BitDP demonstrates consistent effectiveness across InfiniBand and Ethernet network configurations. Across both network environments, BitDP-1bit consistently outperforms both BitDP-2bit and the FP32 gradient baseline. The consistent low standard deviations across all measurements confirm the stability of our approach under real distributed training workloads.

Method	Network	Tokens/s	Speedup
baseline	InfiniBand	81.5 ± 1.5	1.00×
BitDP-2bit	InfiniBand	241.0 ± 7.7	2.96×
BitDP-1bit	InfiniBand	420.0 ± 8.6	5.15×
baseline	Ethernet	2.56 ± 0.01	1.00×
BitDP-2bit	Ethernet	10.01 ± 0.03	3.91×
BitDP-1bit	Ethernet	19.15 ± 0.06	7.47×

Table 5: End-to-end training throughput and speedup ratio for GPT-13B model. Experiments are performed using 64 GPUs configured with data parallelism degree of 8 and tensor parallelism degree of 8.

The practical benefits of BitDP become more pronounced on bandwidth-constrained Ethernet networks, where BitDP achieves 7.47× (1-bit) and 3.91× (2-bit) speedups, compared to 5.15× (1-bit) and 2.96× (2-bit) speedups on high-bandwidth InfiniBand networks as shown in Table 5. These higher speedups result from quantization effectively alleviating the communication bottleneck that typically limits training efficiency in such environments. The consistent pattern where Ethernet configurations yield higher speedup ratios compared to InfiniBand supports our hypothesis that communication compression provides greater benefits under bandwidth-constrained conditions. These results indicate that BitDP can improve training performance across different network infrastructures, while also enabling effective large-scale model training on more accessible and cost-effective network configurations.

Conclusion

In this paper, we introduced BitDP, a comprehensive ultra-low-bit gradient quantization and communication system that integrates theoretical foundations, algorithmic innovations, and system optimizations for LLM training. Starting from the natural distribution properties of gradients, we derived the optimal solution for ultra-low-bit quantization and developed an efficient ultra-low-bit communication technique specifically designed for modern large-scale model training systems. Our evaluation demonstrates that BitDP reduces communication costs by up to 32× while achieving significant training speedups. These results establish BitDP as a theoretically grounded and practically effective solution that democratizes access to large-scale model training across diverse network infrastructures.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Alistarh, D.; Grubic, D.; Li, J.; Tomioka, R.; and Vojnovic, M. 2017. QSGD: Communication-efficient SGD via gradient quantization and encoding. *Advances in neural information processing systems*, 30.

Bisk, Y.; Zellers, R.; Gao, J.; Choi, Y.; et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 7432–7439.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 1877–1901. Curran Associates, Inc.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2924–2936.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 4171–4186.

Gokaslan, A.; and Cohen, V. 2019. OpenWebText Corpus. http://github.com/jcpeterson/openwebtext. Accessed: 2024-07-06.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Jia, J.; Xie, C.; Lu, H.; Wang, D.; Feng, H.; Zhang, C.; Sun, B.; Lin, H.; Zhang, Z.; Liu, X.; et al. 2024. Sdp4bit: Toward 4-bit communication quantization in sharded data parallelism for LLM training. *Advances in Neural Information Processing Systems*, 37: 8734–8759.

Lin, Y.; Han, S.; Mao, H.; Wang, Y.; and Dally, B. 2018. Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training. In *International Conference on Learning Representations*.

Luo, Y.; Ren, X.; Zheng, Z.; Jiang, Z.; Jiang, X.; and You, Y. 2023. CAME: Confidence-guided Adaptive Memory Efficient Optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 4442–4453.

Narayanan, D.; Shoeybi, M.; Casper, J.; LeGresley, P.; Patwary, M.; Korthikanti, V.; Vainbrand, D.; Kashinkunti, P.; Bernauer, J.; Catanzaro, B.; et al. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–15.

NVIDIA Corporation. 2020. NVIDIA NCCL. https://developer.nvidia.com/nccl. Accessed: 2024-07-06.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Rajbhandari, S.; Rasley, J.; Ruwase, O.; and He, Y. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–16. IEEE.

Roemmele, M.; Bejan, C. A.; and Gordon, A. S. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, 90–95.

Shi, S.; Zhou, X.; Song, S.; Wang, X.; Zhu, Z.; Huang, X.; Jiang, X.; Zhou, F.; Guo, Z.; Xie, L.; et al. 2021. Towards scalable distributed training of deep learning on public cloud clusters. *Proceedings of Machine Learning and Systems*, 3: 401–412.

Shoeybi, M.; Patwary, M.; Puri, R.; LeGresley, P.; Casper, J.; and Catanzaro, B. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Su, J.; Ahmed, M.; Lu, Y.; Pan, S.; Bo, W.; and Liu, Y. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568: 127063.

Team, G.; Anil, R.; Borgeaud, S.; Wu, Y.; Alayrac, J.-B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A. M.; Hauth, A.; et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, G.; Qin, H.; Jacobs, S. A.; Wu, X.; Holmes, C.; Yao, Z.; Rajbhandari, S.; Ruwase, O.; Yan, F.; Yang, L.; et al. 2024. ZeRO++: Extremely Efficient Collective Communication for Large Model Training. In *The Twelfth International Conference on Learning Representations*.

Wen, W.; Xu, C.; Yan, F.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2017. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in neural information processing systems*, 30.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4791–4800.

Zhang, B.; and Sennrich, R. 2019. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32.

Zhang, S.; Liu, X.; Liu, J.; Gao, J.; Duh, K.; and Van Durme, B. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, 19–27.