IT TAKES A GOOD MODEL TO TRAIN A GOOD MODEL: GENERALIZED GAUSSIAN PRIORS FOR OPTIMIZED LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

While the rapid advancement of large language models (LLMs) drives increased scale, data transfer, and computational demands, the statistical structures of their weights, activations, and gradients—and their implications for initialization, training dynamics, and efficiency—remain largely unexplored. We empirically show that these quantities in LLMs are well modeled by generalized Gaussian (GG) distributions, and introduce a unified, end-to-end optimization framework grounded in this observation. Our contributions are threefold: (1) a GG-based initialization that aligns with trained model statistics, accelerating training and improving accuracy; (2) activation constrained training (ACT), which regularizes activation representations with GG constraints, reducing pipeline-parallel transmission overhead; and (3) gradient constrained training (GCT) that enforces low-bitrate gradient structures, enhancing communication efficiency in distributed training. Experiments across diverse architectures demonstrate consistently smaller, faster models with minimal communication overhead that match or surpass standard baselines. By anchoring LLM optimization in principled statistical modeling, this work advances efficient, scalable, and hardware-aware AI systems.

1 INTRODUCTION

Large language models (LLMs) achieve impressive performance across various tasks such as code generation, question answering, and reasoning, while their substantial internal representations (e.g., weights, activations, and gradients) impose severe challenges for training, deployment, and inference under resource constraints. Many post-training algorithms, including quantization, sparsification, and distillation, are proposed to reduce element redundancy in precision, structure, and magnitude. Nevertheless, their achievable compression is fundamentally constrained by the Shannon entropy of the data distribution shown in Table 1, which is largely overlooked in prior work. A more fundamental question therefore raised: what statistical distributions govern internal representations in LLMs, and whether this structure can be exploited to enable a more efficient and effective model training and convergence?

Recently, BackSlash (Wu et al., 2025) introduced a constrained training method that optimizes weight entropy during training time, thereby improving compressibility and the effectiveness of pruning and entropy coding with minimal performance loss. Notably, model weights were observed to eventually converge to a Generalized Gaussian Distribution (GGD). Nevertheless, this GG characteristic has received limited attention in BackSlash, and it mainly exhibits two limitations: (1) Despite assuming a GGD prior for weights, BackSlash still trains models from Gaussian initialization, leaving the GG initialization largely unexplored, which may provide better performance and bitrate for models. (2) While BackSlash's GG observation and constrained training scheme are restricted to weights, similar structures and optimizations may also exist in activations and gradients, which may improve the efficiency and compressibility for distributed training.

In this paper, we extensively integrate the GG model to achieve distribution-aware LLM training, rather than solely as an observation. We initialize model weights directly from low-entropy GGDs, building on the GG prior of weights in trained models, promoting inherently lower bitrates and improved convergence behavior. We further discover that activations and gradients exhibit even

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: GG entropy of BERT weights under Quantization (step 2^{-8}), Sparsification (ℓ_1-based 50% sparsity), Distilation (BERT-Large to BERT-Base), and Constrained Training ($\lambda = 1000$). These are the most extreme compression strategies while preserving the original accuracy, but the entropy of the model remains almost unchanged except for Constrained Training.

Method	Quantization	Sparsification	Distillation	Constrained Training
Original Entropy	3.64	3.64	3.64	3.64
Compressed Entropy	3.80	3.40	3.80	**1.85**

stronger GGD characteristics than weights. Guided by this observation, we extend constrained optimization to activation and gradient, and introduce corresponding constrained training algorithms that enable LLMs to produce naturally more compressible representations with slight and controlled performance loss throughout training.

Our main contributions are summarized as follows:

- **Generalized Gaussian Structure of LLMs.** We demonstrate that weights, activations, and gradients in LLMs consistently follow generalized Gaussian distributions, with shape parameters typically below 2, especially for activations and gradients.
- **GG-Guided Initialization and Constrained Training.** We propose a GGD-aligned initialization scheme and distribution-constrained optimization to activations and gradients, enabling faster training, improved generalization, and reduced communication.
- **End-to-End Framework for Efficient LLMs.** We introduce a unified GG-based framework spanning initialization, training, communication, and inference, yielding smaller, faster models with reduced overhead while matching or exceeding standard baselines across architectures and benchmarks.

2 Related Work

2.1 Training Initialization for LLMs

Proper initialization is crucial for training LLMs, as it directly affects convergence speed, generalization, and training stability. Classical schemes—including Gaussian, Xavier (Glorot & Bengio, 2010), and He initialization (He et al., 2015)—regulate variance and gradient flow, with Gaussian initialization playing a central role in symmetry breaking. Beyond these general schemes, orthogonal initialization stabilizes variance in recurrent models (Saxe et al., 2014), IDInit preserves identity mappings in residual networks (Pan et al., 2025), and DaWin enables training-free, inference-time adaptation via entropy-driven weight adjustment (Oh et al., 2025). Recent probabilistic analyses revisit initialization statistics (Wolinski & Arbel, 2025; Basteri & Trevisan, 2022) and explore Gaussian mixtures for improved convergence (Shi & Shang, 2024), motivating more expressive, distribution-aware initialization strategies. Together, these works underscore the central role of principled initialization in stable and efficient training.

2.2 Activation and Gradient Compression

Activations and gradients are crucial for model convergence and training quality, but impose substantial communication overhead in pipeline parallelism and data parallelism of distributed training and inference. Efficient compression can mitigate communication bottlenecks while preserving salient information for robust representation learning. Many compression methods for activations and gradients are proposed and mainly include quantization, sparsification, and entropy coding.

For activations, their distributions frequently exhibit pronounced outliers, necessitating mitigation strategies such as clipping, scaling, structural transformations, or training-aware parameter learning (Xiao et al., 2023a; Ashkboos et al., 2024; Esser et al., 2020; Choi et al., 2018). Furthermore, the lower saliency of small-magnitude activations facilitates sparsity through ReLU/MoE architectures, dynamic prediction, or specialized structural transformations (Agarap, 2019; Shazeer et al., 2017; Liu et al., 2023; Mirzadeh et al., 2023; Nikdan et al., 2023). Beyond sparsity, low-rank and encoding

methods exploit the inherent redundancy within activation spaces (Shamshoum et al., 2025; Wang et al., 2021; Jain et al., 2018).

For gradients, quantization evolved from early 1-bit and ternary schemes (Seide et al., 2014; Wen et al., 2017) to unbiased stochastic quantization and error-feedback mechanisms that preserve convergence (Alistarh et al., 2017; Wu et al., 2018; Karimireddy et al., 2019), and further to adaptive strategies Bernstein et al. (2018); Faghri et al. (2020). Sparsification transmits only the most significant gradient components via Top-k, random sampling, or accumulation schemes Aji & Heafield (2017); Basu et al. (2019); Stich et al. (2018); Lin et al. (2018); Song et al. (2021). Hybrid approaches that combine sparsification and quantization have also been proposed to further improve communication efficiency (Jiang & Agrawal, 2018; Xie et al., 2024).

These methods strove to drive activations and gradients toward the entropy limits of their source distributions, making pioneering contributions to efficient training and communication. In this paper, we introduce constrained training on activations and gradients to directly shape lower-entropy representations at the source, enabling seamless integration with existing compression techniques and establishing a more efficient overall compression framework.

2.3 Optimized LLM Training using Exp-Golomb (EG) Codes and GGDs

Wu et al. (2025) introduces a training-time compression framework, named BackSlash, that formulates large model optimization as a rate-distortion problem, rather than treating compression as a post hoc step. It further models LLM weights using the GGD and EG codes for entropy coding and produces models that are both sparse and quantization-friendly. It is reported that this approach can reduce parameter storage by up to 90% while maintaining accuracy across multiple large language model architectures and tasks. Although this work establishes a promising foundation for integrating statistical modeling into the training dynamics of scalable and hardware-efficient models, it leaves many questions unanswered, some of which we are trying to answer in current research.

3 Generalized Gaussian Prior

A GGD can be expressed as

$$f(x;\mu,\beta,\nu) = \frac{\nu}{2\beta\Gamma(1/\nu)} e^{-(\frac{|x-\mu|}{\beta})^{\nu}}, \tag{1}$$

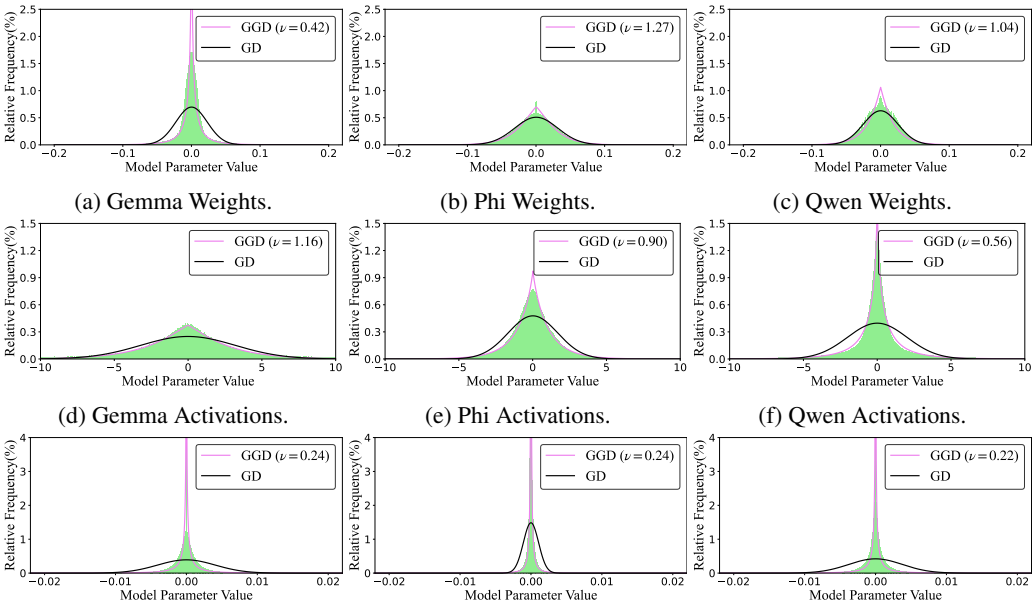
where

$$\beta = \sigma\sqrt{\frac{\Gamma(1/\nu)}{\Gamma(3/\nu)}},\ \Gamma(x) = \int_0^{\infty} t^{x-1}e^{-t}dt, \tag{2}$$

and μ, β,ν are the location, scale, and shape parameters, respectively. Obviously, the Gaussian and Laplacian distributions are special cases of the GGD when $\nu = 2$ and $\nu = 1$. A smaller shape parameter ν corresponds to a higher peak and a heavier tail, and it usually implies a lower differential entropy. Detailed effects of μ, β,ν are described in the Appendix.

Gaussian distributions have long been widely adopted in the modeling and analysis of LLMs (He et al., 2015; Ioffe & Szegedy, 2015; Kingma & Welling, 2013). BackSlash pioneered the GG priors to model the weight distributions of LLMs, demonstrating superior fitting performance over the Gaussian assumption. Building upon this observation, we further conduct validation across more LLMs, including Gemma-3 (Team et al., 2024), Phi-2 (Javaheripi et al., 2023), and Qwen3 (Yang et al., 2025), which reinforces the suitability of the GG prior as a more expressive modeling choice.

Moreover, we further examine the distributions of activations and gradients in these LLMs. The results indicate that they also exhibit GG characteristics with low shape parameters, with even more pronounced than those observed in weight distributions, as illustrated in Fig. 1. A detailed discussion of the GGD distribution is provided in Appendix B. This suggests that the GG prior is widely present in the intrinsic representations of the LLMs, which further motivates us to adopt GGD for weight initialization and to explore constrained training on activations and gradients. The application of the GG prior is reflected in three aspects in this paper:



(a) Gemma Weights. (b) Phi Weights. (c) Qwen Weights.

(d) Gemma Activations. (e) Phi Activations. (f) Qwen Activations.

(g) Gemma Gradients. (h) Phi Gradients. (i) Qwen Gradients.

Figure 1: Statistical distributions of weights, activations, and gradients in full-parameter fine-tuning of certain models, fitted with Generalized Gaussian Distribution (GGD) and Gaussian Distribution (GD), respectively. The statistics for weights and gradients are computed over all trainable parameters, while activations are collected from the output of each Transformer block during forward propagation. It can be observed that GGD achieves better fitting performance than GD in all cases.

- **Entropy Encoding and Communication.** We adopt EG coding, which is well suited for low-shape generalized Gaussian sources (Wen & Villasenor, 1999), especially 0-order EG codes, offering high hardware parallelism and differentiable code length. This enables direct entropy regularization during training and ensures consistency between training-time and communication bitrates.
- **GG-Based Weight Initialization.** We initialize weights from low-entropy generalized Gaussian distributions, yielding models that retain lower entropy and improved compressibility after training.
- **Activation and Gradient Bitrate Control.** Leveraging the strong GG characteristics of activations and gradients, we incorporate average EG code length into the loss to enforce bitrate constraints throughout training, aligning with practical communication.

4 GG IN LLM TRAINING

4.1 GG INITIALIZATION (GG INIT)

Recent work Wu et al. (2025) shows that weights of various LLMs are well modeled by GGDs. While BackSlash formulates LLM training as a rate–distortion optimization using a GGD prior, it still relies on Gaussian initialization, despite parameters ultimately converging to GGDs. This suggests that directly initializing weights from GGDs may accelerate convergence, improve performance, or both.

Consider a linear layer with input $X^{n\times d_{in}}$, output $Y^{n\times d_{out}}$, and weight matrix $W^{d_{in}\times d_{out}}$. Following standard initialization analysis, we assume $x \in X$, $y \in Y$, and $w \in W$ are i.i.d., zero-mean GGD variables, with x and w independent and sharing common variances. Under these assumptions, the variance of a GGD is given by

$$\mathbb{D}[x] = \int_{-\infty}^{+\infty} x^2 f(x; 0, \beta, \nu) dx = \beta^2 \frac{\Gamma(3/\nu)}{\Gamma(1/\nu)}. \tag{3}$$

According to the variance invariance assumption of the input and output data, we have the $\mathbb{D}[x] = \mathbb{D}[y]$. and for the k-th dimension $y_{i,k} \in Y$ of any sample y_i, $y_{i,k} = \sum_{j=1}^{d_{\text{in}}} w_{j,k} x_{i,j}$, so $\mathbb{D}[y_{i,k}] = \mathbb{D}\left[\sum_{j=1}^{d_{\text{in}}} w_{j,k} x_{i,j}\right]$. Since x, y, and w are i.i.d., and w is independent of x, it follows that $\mathbb{D}[y] = d_{\text{in}} \cdot \mathbb{D}[w] \cdot \mathbb{D}[x]$, and therefore

$$\beta_w = \sqrt{\frac{1}{d_{in}} \cdot \frac{\Gamma(1/\nu_w)}{\Gamma(3/\nu_w)}}. \tag{4}$$

Consider the effect of different activation functions have on the variance of neuron output distributions, we introduce a correction coefficient ξ to adjust the variance, which is set to 1 for the case without an activation function or with a Sigmoid, $\xi = 2$ for ReLU, and $\xi = \frac{2}{1+k^2}$ for Leaky-ReLU, where k is the slope of the negative half-axis (He et al., 2015). And therefore, GG initialization for linear layers in neural networks

$$W^{d_{in} \times d_{out}} \sim GG\left(0, \sqrt{\frac{\xi}{d_{in}} \cdot \frac{\Gamma(1/\nu)}{\Gamma(3/\nu)}}, \nu\right). \tag{5}$$

The shape parameter ν acts as a hyperparameter in the GG initialization. When $\nu = 2$, GG initialization degenerates to He initialization. Lower shape parameters preliminarily lead to lower entropy at both initialization and after training, which is shown in Table 2.

Table 2: GG Entropy of BERT weights under GG Init with different shapes, before and after training.

Shape Parameter	**0.1**	**0.5**	**1.0**	**2.0**
Initial Entropy	**2.01**	3.35	3.57	3.66
Trained Entropy	**1.96**	3.35	3.57	3.89

4.2 Activation Constrained Training (ACT)

Based on rate-distortion optimization (RDO) theory (Shannon, 1993), the training objective $\mathcal{J}_{\text{act}}(w)$ jointly optimizing model performance and activation bitrate can be expressed as

$$\mathcal{J}_{\text{act}}(w) = \mathcal{D}(X, Y; w) + \lambda_{\text{act}} \cdot \sum_{i=1}^{n_c} \mathcal{R}(A_i), \tag{6}$$

where X, Y, and w denote the model input, output, and weights, respectively. The model is assumed to be partitioned into $n_c + 1$ modules across different devices, and each module produces an activation communicated to the subsequent module in forward propagation, denoted by $\{A_i\}_{i=1}^{n_c}$. $\mathcal{D}(\cdot)$ is the distortion function, such as MSE, cross-entropy, etc. $\mathcal{R}(\cdot)$ is the rate function of the activations output by various modules, implemented by average EG code length as discussed in Section 3. λ_{act} is the Lagrange multiplier, used to achieve a trade-off between performance and compression. Notably, when $\lambda_{\text{act}} = 0$, ACT will degenerate to conventional training.

The detailed computation process of the rate function $\mathcal{R}(\cdot)$ is presented as follows. Firstly, activations need to be linearly quantized with step 2^{-n} for entropy coding. Notably, straight-through estimator (Liu et al., 2022) $\text{STE}(x) = x - \text{sg}(x - \lfloor x \rfloor)$ is adopted to decouple forward quantization from backward gradients. For n_i-dimension activation vector $A_i \in \mathbb{R}^{n_i}$, the quantization is formulated as

$$A_i^{(q)} = \text{Quant}(A_i) = \text{STE}(2^n \cdot A_i). \tag{7}$$

Then $A_i^{(q)}$ needs to be applied to a non-negative mapping to facilitate subsequent encoding:

$$A_i^{(q,m)} = \text{Map}(A_i^{(q)}) = \begin{cases} -2A_i^{(q)} & A_i^{(q)} < 0 \\ 0 & A_i^{(q)} = 0 \\ 2A_i^{(q)} - 1 & A_i^{(q)} > 0 \end{cases}. \tag{8}$$

The unsigned $A_i^{(q,m)} = \text{Map}(\text{Quant}(A_i))$ can then be directly encoded into code lengths based on element values using 0-order EG encoding. The rate function $\mathcal{R}(\cdot)$ can be derived as

$$\mathcal{R}(A_i) = \frac{2}{n_i} \sum_{j=1}^{n_i} \log_2 \left(A_{i,j}^{(q,m)} + 1 \right) + 1. \tag{9}$$

The training workflow of the ACT algorithm is outlined in Algorithm 1.

Algorithm 1 Activation Constrained Training (ACT)

Require: Distortion function $\mathcal{D}$, model weights w, dataset $\{X, Y\}$, learning rate η, Lagrange multiplier λ_{act}, quantization step 2^{-n}, n_c for number of model shards.
for each epoch $\tau = 0, 1, 2, \dots$ **do**
  for each batch $\{x_b, y_b\}$ sampled from $\{X, Y\}$ **do**
    Forward and calculate distortion $\mathcal{D}(x_b, y_b; w)$ and activations $\{A_1, A_2, \cdots, A_{n_c}\}$.
    Calculate activation rate $\mathcal{R}(A_i) \leftarrow \frac{2}{n_i} \sum_{j=1}^{n_i} \log_2(\text{Map}\left(\text{Quant}(A_{i,j})\right) + 1) + 1$.
    Compute the RD Cost $\mathcal{J}_{act}(w) \leftarrow \mathcal{D}(x_b, y_b; w) + \lambda_{act} \cdot \sum_{i=1}^{n_c} \mathcal{R}(A_i)$.
    Back propagation and update the weights $w \leftarrow w - \eta \cdot \frac{\partial \mathcal{J}_{act}(w)}{\partial w}$.
  end for
end for
Until convergence or max iterations.

4.3 Gradient Constrained Training (GCT)

Constrained training for gradients may be more complex. Assuming gradient $G = \frac{\partial \mathcal{D}(X,Y;w)}{\partial w}$, the optimization objective $\mathcal{J}_{\text{grad}}(w)$ can be similarly expressed as

$$\mathcal{J}_{grad}(w) = \mathcal{D}(X, Y; w) + \lambda_{\text{grad}} \cdot \mathcal{R}(G), \tag{10}$$

where X, Y, and w represent the input, output, and weights. And $\mathcal{D}(\cdot)$, $\mathcal{R}(\cdot)$, λ_{grad} denote the distortion, rate, and Lagrange multiplier. $\mathcal{R}(\cdot)$ also uses average EG code length for implementation. We directly present the weight update to increase the parallelism

$$w_i^{(t+1)} = w_i^{(t)} - \eta \cdot \left(\frac{\partial \mathcal{D}^{(t)}}{\partial w_i^{(t)}} + \lambda_{\text{grad}} \cdot \frac{\partial \mathcal{R}^{(t)}}{\partial w_i^{(t)}} \right), \tag{11}$$

where $\frac{\partial \mathcal{R}^{(t)}}{\partial w_i^{(t)}} = \sum_{j=1}^{n_{\text{grad}}} \rho_j^{(t)} \cdot \frac{\partial^2 \mathcal{D}^{(t)}}{\partial w_j^{(t)} \partial w_i^{(t)}}$. Here $\rho_j^{(t)} = \frac{\partial \mathcal{R}^{(t)}}{\partial g_j^{(t)}} \approx \frac{2}{n_{\text{grad}} \ln 2} \cdot \frac{\text{Sign}(g_j^{(t)})}{|g_j^{(t)}| + \epsilon_0}$ is the rate gradient factor, where n_{grad} is gradient number and ϵ_0 is a smoothing constant.

Computing the full Hessian matrix is computationally and memory-prohibitive, so we employ a diagonal approximation (Sutskever et al., 2013; Bottou et al., 2018; Neal & Hinton, 1998) that retains only $\frac{\partial^2 D}{\partial w_i^2}$, under which Equation 11 simplifies to

$$w_i^{(t+1)} = w_i^{(t)} - \eta \left(\frac{\partial \mathcal{D}^{(t)}}{\partial w_i^{(t)}} + \lambda_{\text{grad}} \cdot \rho_i^{(t)} \cdot \frac{\partial^2 \mathcal{D}^{(t)}}{\partial w_i^{(t)2}} \right). \tag{12}$$

We further apply an efficient approximation to $\frac{\partial^2 D}{\partial w_i^2}$. Let $h_i^{(t)} = \frac{\partial^2 \mathcal{D}^{(t)}}{\partial w_i^{(t)2}}$, $g_i^{(t)} = \frac{\partial \mathcal{D}^{(t)}}{\partial w_i^{(t)}}$, and $\hat{g}_i^{(t)} = g_i^{(t)} + \lambda_{\text{grad}} \cdot \rho_i^{(t)} \cdot h_i^{(t)}$, so that the weight update reads $w_i^{(t)} = w_i^{(t-1)} - \eta \cdot \hat{g}_i^{(t-1)}$. With the small weight difference $\delta w = w_i^{(t)} - w_i^{(t-1)} = -\eta \cdot \hat{g}_i^{(t-1)}$, $h_i^{(t)}$ can be approximated as

$$\hat{h}_i^{(t)} = \frac{g_i^{(t)} - g_i^{(t-1)}}{w_i^{(t)} - w_i^{(t-1)}} = -\frac{g_i^{(t)} - g_i^{(t-1)}}{\eta \cdot \hat{g}_i^{(t-1)}}. \tag{13}$$

In practice, we approximate $g_i^{(t-1)}$ with $\hat{g}_i^{(t-1)}$, as a small λ_{grad} makes them numerically close, eliminating the storage of $g_i^{(t)}$. Moreover, $\hat{g}_i^{(t-1)}$ induces a momentum compensation term $\hat{h}_i^{(t-1)}$

for $\hat{h}_i^{(t)}$, thereby stabilizing optimization. With a smoothing constant ϵ for soft gradient clipping, Equation 13 can be rewritten as

$$\hat{h}_i^{(t)} = -\frac{1}{\eta} \cdot \frac{g_i^{(t)}}{\hat{g}_i^{(t-1)} + \epsilon} + \frac{1}{\eta}, \tag{14}$$

which requires storing only the previous-step $\hat{g}_i^{(t-1)}$ to approximate second-order derivatives. Under this approximation, the weight update in Equation 11 simplifies to

$$w_i^{(t+1)} = w_i^{(t)} - \eta \cdot (g_i^{(t)} + \lambda_{\text{grad}} \cdot \rho_i^{(t)} \cdot \hat{h}_i^{(t)}). \tag{15}$$

To improve convergence in late training, we further apply a decaying Lagrange multiplier updated as $\lambda_{\text{grad}}^{(\tau+1)} = \alpha \cdot \lambda_{\text{grad}}^{(\tau)}$ with $0 < \alpha < 1$.

The GCT algorithm can be summarized in Algorithm 2:

Algorithm 2 Decaying Gradient Constrained Training Algorithm (GCT)

Require: Distortion function $\mathcal{D}$, weights w, dataset $\{X, Y\}$, global step $t = 1$, learning rate η, Lagrange multiplier λ_{grad}, smoothing constant ϵ, decaying coefficient α.
for each epoch $\tau = 0, 1, 2, \ldots$ **do**
  for each batch $\{x_b, y_b\}$ sampled from $\{X, Y\}$ **do**
    Forward and backward propagation to compute the gradients: $g_i^{(t)} \leftarrow \partial \mathcal{D}^{(t)} / \partial w_i^{(t)}$.
    Compute approximate second-order derivatives: $\hat{h}_i^{(t)} \leftarrow (-\frac{1}{\eta} \cdot \frac{g_i^{(t)}}{\hat{g}_i^{(t-1)} + \epsilon} + \frac{1}{\eta}) \cdot \mathbf{1}_{t \geq 2}$.
    Compute and store the equivalent gradient: $\hat{g}_i^{(t)} \leftarrow g_i^{(t)} + \lambda_{\text{grad}}^{(\tau)} \cdot \rho_i^{(t)} \cdot \hat{h}_i^{(t)}$.
    Weight update: $w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta \cdot \hat{g}_i^{(t)}$, $t \leftarrow t + 1$.
  end for
  Constraint decay: $\lambda_{\text{grad}}^{(\tau+1)} = \alpha \cdot \lambda_{\text{grad}}^{(\tau)}$.
end for
Until convergence or max iterations.

5 Experiments

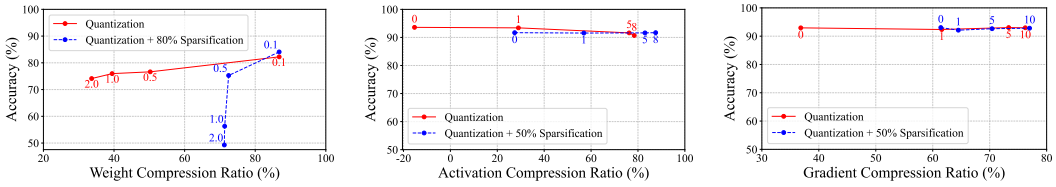
5.1 Settings

GG Init, ACT, and GCT are evaluated through extensive experiments across diverse models and tasks, analyzing training dynamics, generalization, and deployment efficiency. Experiments span models on BERT (Devlin et al., 2019), Qwen3 (Yang et al., 2025), Gemma-3 (Team et al., 2024) and DeepSeek (Guo et al., 2025), trained with tasks on IMDB (Maas et al., 2011), QNLI (Wang et al., 2018), WebQuestions (Berant et al., 2013), and SQuAD (Rajpurkar et al., 2016),respectively. Performance is measured by accuracy or next-token accuracy, while compression is quantified by average code lengths with 0-order EG and Huffman (HM) coding against Fixed-Length (FL) coding, following (Wu et al., 2025), where each scheme is linearly quantized with the same quantization step before encoding. C_{EG} and C_{HM} denote compression ratios relative to FL.

Complete training, encoding, and hardware configurations are described in Appendix D.1. System-level measurements of ACT and GCT, together with the synergy between GG Init and BackSlash training, are provided in Appendix D. Detailed analyses of GGDs are summarized in Appendix B.

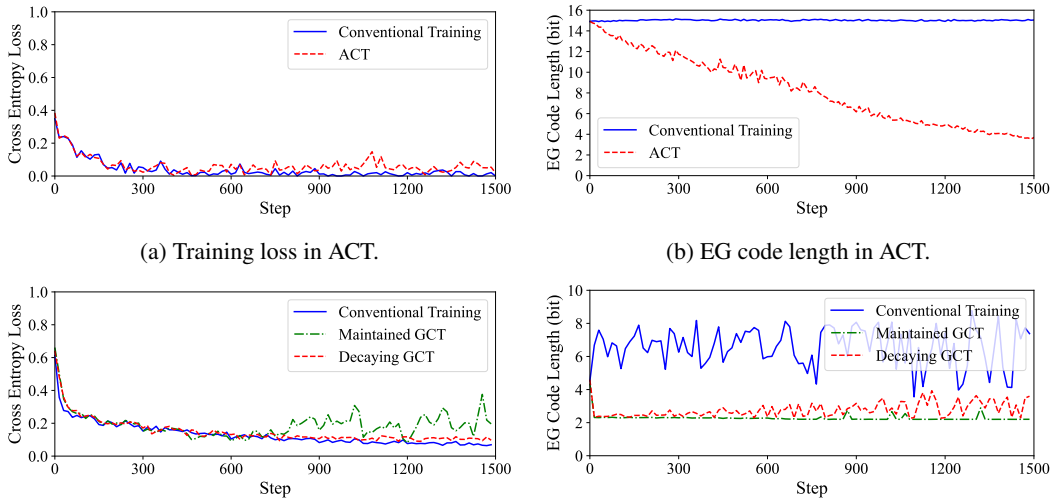
5.2 Trade-off

We analyze the performance-bitrate trade-off to assess the robustness of hyperparameter selection for GG Init, ACT, and GCT. Using BERT on IMDB, we vary the GG Init shape parameter and the ACT/GCT Lagrange multipliers to obtain trade-off curves. As shown in Fig. 2, GG Init exhibits a synchronous increase in accuracy and bitrate, suggesting that lower shapes ease optimization and improve generalization. ACT and GCT exhibit an inverse trade-off that aligns with the RDO theory,



(a) GG Initialization varying ν. (b) ACT varying λ_{act} ($\times 10^{-3}$). (c) GCT varying λ_{grad} ($\times 10^{-6}$).

Figure 2: The trade-off between accuracy and EG compression rate by varying shape parameters in GG Init, Lagrange multipliers in ACT, and GCT. Specifically, $\nu = 2.0$ in GG Init and $\lambda = 0$ in ACT/GCT degenerate into Gaussian initialization and conventional training, serving as controls.



(a) Training loss in ACT. (b) EG code length in ACT.

(c) Training loss in GCT. (d) EG code length in GCT.

Figure 3: The training loss and average EG code length for ACT ($\lambda_{\text{act}} = 1 \times 10^{-3}$), GCT ($\lambda_{\text{grad}} = 1 \times 10^{-5}$), compared to conventional training ($\lambda_{\text{act}} = \lambda_{\text{grad}} = 0$).

whose accuracy remains stable as bitrate decreases, demonstrating that ACT and GCT can significantly reduce communication cost while preserving performance. Additionally, we further study their synergy with linear ℓ_1 sparsification. GG Init ($\nu = 0.1$) further improves both compression and performance when Gaussian initialization ($\nu = 2.0$) nearly fails, and ACT/GCT achieves better compression across all groups with almost no performance loss. These results demonstrates that these distribution-aware compression methods are complementary to quantization and sparsification, and can be effectively combined with them.

5.3 Training Performance

As ACT and GCT regulate dynamically generated activations and gradients, tracking loss and bitrate during training reveals their underlying behaviors. Using BERT on IMDB, we compare ACT and GCT to conventional training by monitoring loss and EG code length. For ACT, Fig. 3a shows loss curves nearly identical to conventional training, indicating unaffected convergence. Conversely, Fig. 3b shows ACT progressively reduces activation bitrate, while conventional training remains high, reflecting ACT's indirect but effective shaping of activation distributions. For GCT, Fig. 3c demonstrates that decaying GCT ($\alpha = 0.8$) closely follows conventional training, whereas non-decay GCT ($\alpha = 1.0$) suffers late-stage oscillations. Fig. 3d further shows that decaying GCT yields a lower and smoother gradient bitrate than conventional training, which exhibits large fluctuations. These results show that ACT and GCT substantially reduce communication cost with stable convergence, highlighting the importance of distribution-aware optimization.

5.4 GENERALIZATION

We evaluate the generalization of GG Init, ACT, and GCT across multiple architectures and tasks using the same hyperparameters for fair comparisons, which can be further adjusted to achieve another trade-off guided by Section 5.2. As shown in Table 3, GG Init consistently achieves higher compression and better accuracy. Generative tasks exhibit higher final weight code lengths than classification tasks, reflecting their greater modeling complexity. ACT and GCT substantially compress activations and gradients with minimal accuracy loss. Notably, ACT produces significantly lower EG code lengths than FL, while conventional training often exceeds FL due to activation outliers (Xiao et al., 2023b), which confirms that ACT effectively reshapes activation distributions and suppresses outliers. Overall, these results demonstrate that GGI, ACT, and GCT generalize robustly across models and tasks, from classification to generation.

Table 3: The generalization performance of GG Init ($\nu = 0.1$), ACT ($\lambda_{\text{act}} = 1 \times 10^{-3}$), and GCT ($\lambda_{\text{grad}} = 1 \times 10^{-5}$) as constrained strategies, compared to Gaussian initialization ($\nu = 2.0$) and conventional training ($\lambda_{\text{act}} = \lambda_{\text{grad}} = 0$) as unconstrained strategies, respectively.

Model	Method	Strategy	EG	HM	FL	C_{EG}	C_{HM}	Accuracy
BERT-0.1B	GG Init	Unconstrained	6.63	5.32	10.00	34%	47%	74.12%
		Constrained	1.59	1.57	12.00	**87%**	**87%**	**82.23%**
	ACT	Unconstrained	14.99	9.53	13.00	-15%	27%	**93.59%**
		Constrained	4.31	4.09	13.00	**67%**	**69%**	91.62%
	GCT	Unconstrained	7.58	6.18	11.00	31%	44%	92.94%
		Constrained	3.43	3.16	12.00	**71%**	**74%**	**92.97%**
Qwen-0.6B	GG Init	Unconstrained	6.38	4.93	11.00	42%	55%	53.18%
		Constrained	2.81	2.69	11.00	**74%**	**76%**	**61.71%**
	ACT	Unconstrained	18.29	11.31	15.00	-22%	25%	89.60%
		Constrained	7.42	6.13	15.00	**51%**	**59%**	**89.75%**
	GCT	Unconstrained	3.93	3.57	12.00	67%	70%	**92.60%**
		Constrained	3.43	2.89	12.00	**71%**	**76%**	91.94%
Gemma-2B	GG Init	Unconstrained	5.50	4.42	10.00	45%	56%	38.83%
		Constrained	3.55	3.25	11.00	**68%**	**70%**	**39.16%**
	ACT	Unconstrained	15.70	10.00	12.00	-31%	17%	**65.03%**
		Constrained	7.16	6.15	11.00	**35%**	**44%**	64.12%
	GCT	Unconstrained	5.07	4.20	12.00	58%	65%	67.66%
		Constrained	2.06	1.57	12.00	**83%**	**87%**	**68.05%**
DeepSeek-7B	GG Init	Unconstrained	5.26	4.16	10.00	47%	58%	25.18%
		Constrained	4.29	3.50	11.00	**61%**	**68%**	**25.30%**
	ACT	Unconstrained	17.46	10.74	12.00	-46%	11%	**70.79%**
		Constrained	7.72	5.95	12.00	**36%**	**50%**	70.67%
	GCT	Unconstrained	4.66	4.24	12.00	61%	65%	45.10%
		Constrained	2.47	1.94	12.00	**79%**	**84%**	**45.31%**

6 CONCLUSION

In this paper, we present a unified, statistically grounded framework for LLM optimization, including GG Init, ACT, and GCT, which leverages GG priors across initialization, training, communication, and inference. By modeling weights, activations, and gradients as heavy-tailed GG distributions, our approach improves convergence, compressibility, and communication efficiency. This framework optimizes the model's statistical distribution during training to ensure low-entropy representations, facilitating and cooperating with quantization, sparsification, and entropy coding. Extensive experiments show consistent gains in compression and efficiency with minimal accuracy loss. This work highlights the value of rate-aware, statistically principled model design and opens avenues for extending GG-informed optimization to federated, and energy-efficient AI systems.

AI USE STATEMENT

Generative AI tools were used only for language polishing of the paper. They were not used to generate experimental data, develop theoretical models or mathematical claims, formulate proofs or hypotheses, design experiments or research methods, or implement training methods. The authors checked the plotted values and structural masks against the evaluation records and take responsibility for the cited references and all content and claims in this work.

REFERENCES

Abien Fred Agarap. Deep learning using rectified linear units (relu), 2019. URL https://arxiv.org/abs/1803.08375.

Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. *arXiv preprint arXiv:1704.05021*, 2017.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

Tiago A Almeida, José María G Hidalgo, and Akebo Yamakami. Contributions to the study of sms spam filtering: new collection and results. In *Proceedings of the 11th ACM symposium on Document engineering*, pp. 259–262, 2011.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 100213–100240. Curran Associates, Inc., 2024. doi: 10.52202/079017-3180.

Loïc Barrault, Ondřej Bojar, Marta R Costa-Jussa, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, et al. Findings of the 2019 conference on machine translation (wmt19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pp. 1–61, 2019.

Andrea Basteri and Dario Trevisan. Quantitative gaussian approximation of randomly initialized deep neural networks, 2022.

Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification and local computations. *Advances in Neural Information Processing Systems*, 32, 2019.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1533–1544, 2013.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International conference on machine learning*, pp. 560–569. PMLR, 2018.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks, 2018. URL https://arxiv.org/abs/1805.06085.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pp. 4171–4186, 2019.

Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. In *International Conference on Learning Representations*, 2020.

Fartash Faghri, Iman Tabrizian, Ilia Markov, Dan Alistarh, Daniel M Roy, and Ali Ramezani-Kebrya. Adaptive gradient quantization for data-parallel sgd. *Advances in neural information processing systems*, 33:3174–3185, 2020.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Animesh Jain, Amar Phanishayee, Jason Mars, Lingjia Tang, and Gennady Pekhimenko. Gist: efficient data encoding for deep neural network training. In *Proceedings of the 45th Annual International Symposium on Computer Architecture*, ISCA '18, pp. 776–789. IEEE Press, 2018. ISBN 9781538659847. doi: 10.1109/ISCA.2018.00070. URL https://doi.org/10.1109/ISCA.2018.00070.

Mojan Javaheripi, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio Mendes, W Chen, AD Giorno, R Eldan, S Gopi, S Gunasekar, et al. The surprising power of small language models. In *Conference presentation*, 2023.

Peng Jiang and Gagan Agrawal. A linear speedup analysis of distributed deep learning with sparse and quantized communication. *Advances in neural information processing systems*, 31, 2018.

Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International conference on machine learning*, pp. 3252–3261. PMLR, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7871–7880, 2020.

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SkhQHMW0W.

Zechun Liu, Kwang-Ting Cheng, Dong Huang, Eric P Xing, and Zhiqiang Shen. Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4942–4952, 2022.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, and Beidi Chen. Deja vu: Contextual sparsity for efficient llms at inference time, 2023. URL https://arxiv.org/abs/2310.17157.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea (eds.), *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015/.

Seyed Iman Mirzadeh, Keivan Alizadeh-Vahid, Sachin Mehta, Carlo C del Mundo, Oncel Tuzel, Golnoosh Samei, Mohammad Rastegari, and Mehrdad Farajtabar. Relu strikes back: Exploiting activation sparsity in large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pp. 355–368. Springer, 1998.

Mahdi Nikdan, Tommaso Pegolotti, Eugenia Iofinova, Eldar Kurtic, and Dan Alistarh. SparseProp: Efficient sparse backpropagation for faster training of neural networks at the edge. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 26215–26227. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/nikdan23a.html.

Changdae Oh, Yixuan Li, Kyungwoo Song, Sangdoo Yun, and Dongyoon Han. Dawin: Training-free dynamic weight interpolation for robust adaptation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=L8e7tBf4pP.

Yu Pan, Chaozheng Wang, Zekai Wu, Qifan Wang, Min Zhang, and Zenglin Xu. Idinit: A universal and stable initialization method for neural network training, 2025. URL https://arxiv.org/abs/2503.04626.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 2383–2392, 2016.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, 2014. URL https://arxiv.org/abs/1312.6120.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns. In *Interspeech*, volume 2014, pp. 1058–1062. Singapore, 2014.

Yara Shamshoum, Nitzan Hodos, Yuval Sieradzki, and Assaf Schuster. Compact: Compressed activations for memory-efficient llm training, 2025. URL https://arxiv.org/abs/2410.15352.

Claude E. Shannon. *Coding Theorems for a Discrete Source With a Fidelity CriterionInstitute of Radio Engineers, International Convention Record, vol. 7, 1959.*, pp. 325–350. 1993. doi: 10.1109/9780470544242.ch21.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Xiao Shi and Yun Shang. Avoiding barren plateaus via gaussian mixture model, 2024. URL https://arxiv.org/abs/2402.13501.

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*, pp. 1631–1642, 2013.

Liuyihan Song, Kang Zhao, Pan Pan, Yu Liu, Yingya Zhang, Yinghui Xu, and Rong Jin. Communication efficient sgd via gradient sampling with bayes prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12065–12074, 2021.

Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in neural information processing systems*, 31, 2018.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/sutskever13.html.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Hongyi Wang, Saurabh Agarwal, and Dimitris Papailiopoulos. Pufferfish: Communication-efficient models at no extra cost, 2021. URL https://arxiv.org/abs/2103.03936.

Jiangtao Wen and J.D. Villasenor. Structured prefix codes for quantized low-shape-parameter generalized gaussian sources. *IEEE Transactions on Information Theory*, 45(4):1307–1314, 1999. doi: 10.1109/18.761289.

Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in neural information processing systems*, 30, 2017.

Pierre Wolinski and Julyan Arbel. Gaussian pre-activations in neural networks: Myth or reality?, 2025. URL https://arxiv.org/abs/2205.12379.

Jiaxiang Wu, Weidong Huang, Junzhou Huang, and Tong Zhang. Error compensated quantized sgd and its applications to large-scale distributed optimization. In *International conference on machine learning*, pp. 5325–5333. PMLR, 2018.

Jun Wu, Jiangtao Wen, and Yuxing Han. BackSlash: Rate constrained optimized training of large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67852–67863. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/wu25ai.html.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 23–29 Jul 2023a. URL https://proceedings.mlr.press/v202/xiao23c.html.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023b.

Weiying Xie, Haowei Li, Jitao Ma, Yunsong Li, Jie Lei, Donglai Liu, and Leyuan Fang. Jointsq: Joint sparsification-quantization for distributed learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5778–5787, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

A Constrained Training and Conventional Compression

Within each training iteration, activations, gradients, and weights are sequentially yielded during the forward pass, backward propagation, and parameter update. Conventional compression techniques, such as quantization, sparsification, and entropy coding, are applied after the training step, and are thus categorized as **post-training** compression. At this stage, the statistical distributions of these tensors are established, and subsequent compressions operate solely on individual elements, rendering them **element-aware** methods. In contrast, constrained training methods, including ACT, GCT, and BackSlash, explicitly reshape these distributions during training to promote low-entropy representations and enhance compressibility. Therefore, they can be viewed as **training-time** and **distribution-aware** compression. As these two paradigms operate at different stages and aspects, they are inherently complementary rather than conflicting, as discussed at the end of Section 5.2.

We illustrate this with an example. A tensor produced by conventional training may take the form $[-0.63, 0.28, \ldots, 0.93, -0.05, 0.36]$, while constrained training yields a lower-entropy counterpart such as $[-0.13, 0.01, \ldots, -1.56, 0.01, 0.02]$. Post-training compression can then be applied in a pipeline manner. For instance, linear quantization with a step size of 0.1 will produce $[-0.1, 0.0, \ldots, -1.6, 0.0, 0.0]$, followed by ℓ_1-based sparsification (threshold 0.2), resulting in $[0.0, 0.0, \ldots, -1.6, 0.0, 0.0]$. Entropy coding can be further applied if needed. This workflow demonstrates that constrained training, quantization, pruning, and entropy coding are mutually compatible and can be sequentially integrated to collectively enhance compression efficiency.

We introduce several classic and well-established conventional compression techniques to further verify their synergy with constrained training. Specifically, NF4 (Dettmers et al., 2023) quantization and Wanda (Sun et al., 2023) sparsification are applied to weights training BERT on IMDB to study their compatibility with GG Init; LSQ (Esser et al., 2020) quantization and ReLU-based (Mirzadeh et al., 2023) sparsification to activations with ACT; and QSGD (Alistarh et al., 2017) quantization and Top-K (Lin et al., 2018) sparsification to gradients with GCT. Compression is measured by the reduction in Exponential-Golomb (EG) and Huffman (HM) code lengths relative to the FP32 baseline, while performance is evaluated by accuracy (Acc). As shown in Fig. 4, GG Init, ACT, and GCT consistently lead to higher compression ratios compared to unconstrained training, and exhibit strong synergy with various quantization and sparsification methods. Meanwhile, ACT and GCT introduce negligible degradation, while GG Init even further improves accuracy and shows robustness, where Gaussian Initialization fails under extreme sparsification.

B Generalized Gaussian Distributions (GGDs)

B.1 Entropy Analysis

From an information-theoretic perspective, analyzing the entropy of the GGD provides a principled characterization of its compressibility. Let the random variable $X \sim \mathrm{GGD}(\mu, \beta, \nu)$, whose probability density function is given by

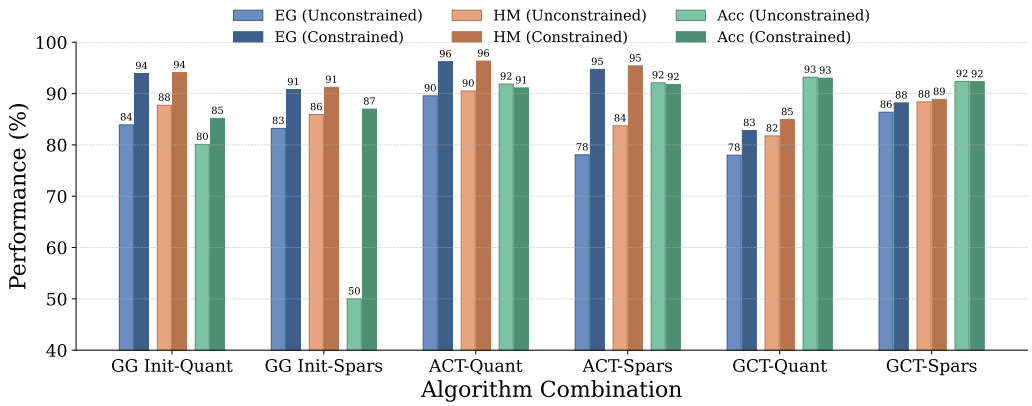
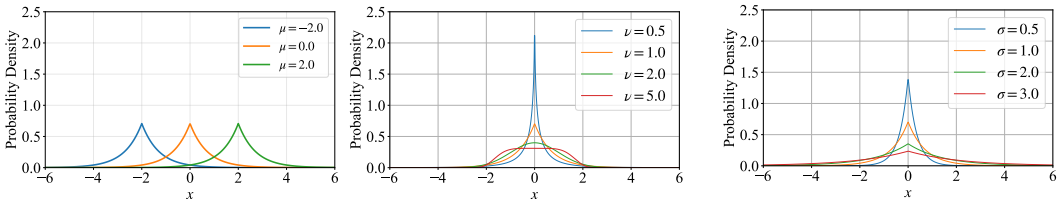


Figure 4: The synergistic effects of constrained training (including GG Init, ACT, and GCT) with quantization and sparsification are evaluated using EG, HM compression, and accuracy.

$$f(x;\mu,\beta,\nu) = \frac{\nu}{2\beta\Gamma(1/\nu)} e^{-(\frac{|x-\mu|}{\beta})^{\nu}}. \tag{16}$$

where μ represents the location parameter, while $\beta > 0$ and $\nu > 0$ denote the scale and shape parameters, respectively; $\Gamma(\cdot)$ is the Gamma function. For a fixed shape ν, the scale β is linearly proportional to the standard deviation σ. As variance offers a more intuitive measure of data dispersion and energy distribution than the scale parameter, we characterize the GGD using μ, ν, and σ^2 in our subsequent analysis. The effects of varying these parameters are visualized in Fig. 5.



(a) GGDs with different μ. (b) GGDs with different ν. (c) GGDs with different σ^2.

Figure 5: The effect of different parameter changes on the shape of the GGD. The baseline configuration is $\mu = 0$, $\nu = 1$, $\sigma^2 = 1$, and only the parameter under investigation is varied. The location parameter μ shifts the curve horizontally, the shape parameter ν modifies the peakedness and tails, and the variance σ^2 scales the spread of the GGDs.

The differential entropy of GGD can be expressed as:

$$H(X) = -\int_{-\infty}^{\infty} f(x)\ln f(x)dx = \frac{1}{\nu} + \ln\left[\frac{2\sigma}{\nu}\sqrt{\frac{\Gamma(1/\nu)^3}{\Gamma(3/\nu)}}\right]. \tag{17}$$

Since the entropy expression is independent of the location parameter μ, we omit its discussion here. For the **variance** σ^2, it is evident that

$$\frac{\partial H(X)}{\partial \sigma^2} = \frac{1}{2\sigma^2} > 0. \tag{18}$$

The entropy is monotonically increasing with respect to the variance σ^2, which is consistent with the intuition in compression: a wider amplitude range implies higher coding cost.

For the **shape parameter** ν, we have:

$$\frac{\partial H(X)}{\partial \nu} = \frac{-2\nu - 2 + 3\left[\psi\left(\frac{3}{\nu}\right) - \psi\left(\frac{1}{\nu}\right)\right]}{2\nu^2}. \tag{19}$$

Table 4: Tensor fitting performance of different distributions on the Qwen-4B model.

Metric	Tensor	Gaussian	Laplace	GMM	Student-t	GGD
Held-out NLL	Weight	1.4189	1.4313	1.4132	1.4038	**1.4023**
	Activation	3.377	INF	17.58	**-0.047**	-0.009
	Gradient	1.4522	0.7565	0.6578	**0.4834**	0.5034
KS distance	Weight	0.0217	0.0370	0.0211	0.0068	**0.0065**
	Activation	0.256	0.026	0.101	**0.006**	0.028
	Gradient	0.2149	0.0552	0.0593	**0.0109**	0.0228
Wasserstein-1	Weight	0.0460	0.0990	0.0469	65.87	**0.0148**
	Activation	0.550	0.036	0.036	**0.013**	0.036
	Gradient	0.4694	0.0891	0.1262	1.8e16	**0.0577**

Table 5: Fitted GGD shape parameter ν at sampled Qwen-4B layers across different tensors.

Tensor	L0	L3	L6	L9	L12	L15	L18	L21	L24	L27	L30	L33	Mean	Std.
Weight	1.72	0.99	1.73	1.67	1.68	1.60	1.58	1.58	1.58	1.71	1.74	1.69	**1.61**	0.20
Activation	0.79	0.74	0.68	0.80	0.77	0.70	0.92	0.90	0.90	0.88	0.85	0.92	**0.82**	0.09
Gradient	0.78	0.18	0.52	0.70	0.79	0.70	0.67	0.59	0.60	0.61	0.60	0.52	**0.61**	0.16

where $\Psi(\cdot)$ is the Digamma function.

The sign of the derivative in Equation 19 must be interpreted consistently with the entropy maximum. For example, at $\nu = 1$, using $\psi(3) - \psi(1) = 3/2$ gives $\partial H(X)/\partial\nu = 1/4 > 0$. Thus the entropy increases locally as ν approaches 2 from this point, and the derivative cannot be nonpositive throughout $(0, 2]$.

A global conclusion follows directly from the maximum-entropy theorem. For fixed variance σ^2, the Gaussian distribution uniquely maximizes differential entropy among all continuous distributions. The GGD is Gaussian exactly when $\nu = 2$ (with the scale chosen to give variance σ^2); therefore

$$H_\nu(X) \leq H_2(X) = \tfrac{1}{2}\ln(2\pi e\sigma^2),$$

with equality only at $\nu = 2$. This establishes that the GGD entropy has its global maximum at the Gaussian case. This argument does not claim that entropy is monotone on each entire interval $(0, 2)$ or $(2, \infty)$; the empirical code-length trend for the tested shapes in Table 2 is a separate experimental observation.

B.2 Empirical GGD Fitting Compared to Other Distributions

We fit Gaussian, Laplace, Gaussian mixture (GMM), Student-t, and generalized Gaussian (GGD) distributions separately to each of the 36 Qwen-4B layers. The table reports the mean held-out negative log-likelihood (NLL), Kolmogorov–Smirnov (KS) distance, and Wasserstein-1 distance across layers. Lower values indicate better fit. GGD consistently achieves the best or second-best fitting performance on all metrics, and even when it is second best, its gap to the best result is very limited. At the same time, the numerical behaviour of GGD across all metrics is more stable, and we therefore believe it is better suited as a general-purpose distribution fitting tool.

B.3 Layer-wise GGD Shape Parameters

We also fit a GGD independently to each layer. The shape parameter ν is below 2 for all sampled layers and has a mean below 2 for weights, activations, and gradients, indicating that the heavy-tailed behavior is present within individual layers rather than arising only from mixing layers.

B.4 GG Init Across Seeds and Initializers

We compare GG Init with He, Xavier, Laplace, Student-t (df=3), and GMM initialization on BERT/IMDB over five epochs and five random seeds. GG Init gives the lowest EG code length and highest accuracy, with low variation across runs.

Table 6: Rate–performance comparison of different distribution-based initialization methods over five random seeds on BERT/IMDB.

Metric	GG Init	He	Xavier	Laplace	Student-t	GMM
EG (bit)	**2.83 ± 0.01**	7.48 ± 0.00	6.67 ± 0.00	6.35 ± 0.00	6.18 ± 0.00	6.42 ± 0.00
Accuracy (%)	**87.11 ± 0.55**	81.74 ± 2.46	84.69 ± 0.59	84.75 ± 1.58	84.69 ± 1.83	85.49 ± 0.81

C End-to-end LLM optimization Framework

The overall working principles and mechanisms of GG Init, ACT, and GCT during the distributed training process are visualized in Fig.6. GG Init is first introduced at the initialization stage to ensure that model weights maintain a low bitrate throughout training. During training, ACT and GCT are applied to constrain the bitrates of activations and gradients, respectively, thereby reducing both intra-model and inter-model communication overhead. In the inference stage, ACT preserves low-bit-rate activations, enabling persistent optimization of activation communication in inference scenarios.

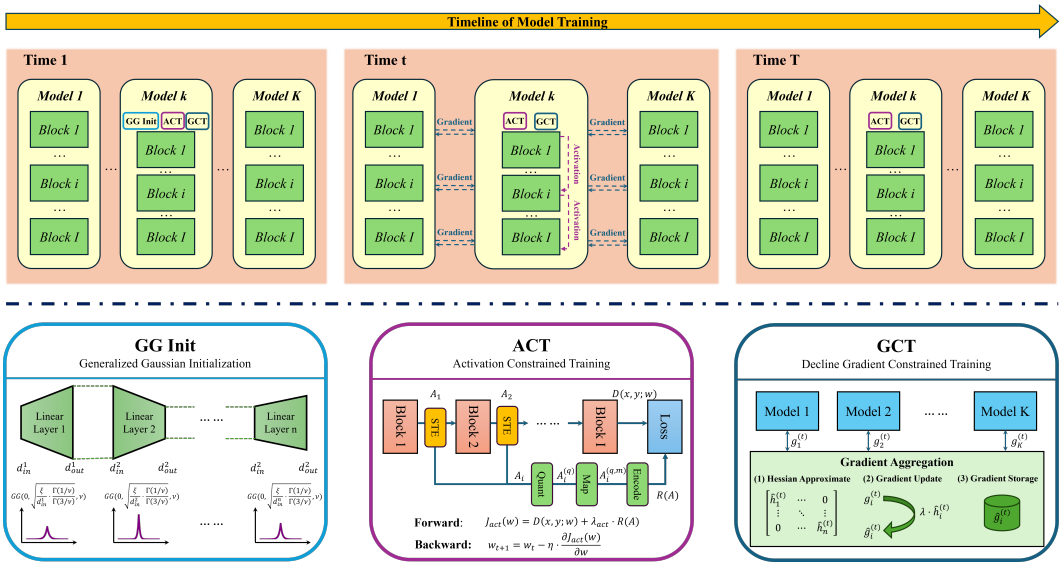


Figure 6: Overview of training process by GG Init, ACT, and GCT. At the initialization phase of training, GG Init is responsible for endowing the model with a low-entropy initial distribution. As training progresses, ACT and GCT are responsible for compressing the gradients generated during forward propagation and the gradients transmitted between models, respectively.

D Supplementary Experiments

D.1 Detailed Settings

We have preliminarily provided the experimental models and datasets, the evaluation methods for performance and compression in Sec. 5.1. Here, we will further present specific details regarding model training, entropy encoding, and hardware configuration.

Training. All models are trained for 20 epochs with a batch size of 16. AdamW optimizer is employed with a constant learning rate of 1×10^{-4} for pretraining and 2×10^{-5} for fine-tuning. The optimizer's hyperparameters were maintained at their standard default values to ensure experimental reproducibility and gradient stability. Specifically, they are set to $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, and a weight decay of 0.01. Input sequences are truncated or padded to 512 tokens. All models are trained with full-parameter optimization.

Performance. We evaluate the model performance using accuracy and next token accuracy. Accuracy is used to evaluate text classification tasks, such as sentiment analysis with BERT on IMDB

Table 7: More generalization performance on compression and performance of GG Init, ACT, and GCT, supplemented in Table 3.

Model	Method	Strategy	EG	HM	FL	C_{EG}	C_{HM}	Accuracy
Bart-0.4B	GG Init	Unconstrained	6.66	5.28	10.00	33%	47%	81.77%
		Constrained	2.67	2.55	11.00	**76%**	**77%**	**82.57%**
	ACT	Unconstrained	12.19	8.27	11.00	-11%	25%	**93.46%**
		Constrained	5.12	4.45	11.00	**53%**	**60%**	92.78%
	GCT	Unconstrained	6.24	5.42	12.00	48%	55%	95.30%
		Constrained	4.79	4.37	12.00	**60%**	**64%**	**96.10%**
GPT-0.8B	GG Init	Unconstrained	5.83	4.72	9.00	35%	48%	**99.25%**
		Constrained	2.16	2.12	11.00	**80%**	**81%**	99.10%
	ACT	Unconstrained	18.22	11.27	15.00	-21%	25%	**99.15%**
		Constrained	3.07	2.64	12.00	**74%**	**78%**	97.60%
	GCT	Unconstrained	6.93	5.84	12.00	42%	51%	**98.05%**
		Constrained	2.97	2.60	12.00	**75%**	**78%**	95.15%
LLaMA-1B	GG Init	Unconstrained	5.63	4.47	9.00	37%	50%	46.13%
		Constrained	4.54	3.55	11.00	**59%**	**68%**	**51.99%**
	ACT	Unconstrained	10.27	7.22	11.00	7%	34%	**70.79%**
		Constrained	4.37	3.89	11.00	**60%**	**65%**	70.67%
	GCT	Unconstrained	6.72	5.58	12.00	44%	54%	**48.44%**
		Constrained	2.75	2.32	12.00	**77%**	**81%**	45.35%
Qwen-4B	GG Init	Unconstrained	5.45	4.31	10.00	46%	57%	**32.46%**
		Constrained	4.13	3.38	11.00	**62%**	**69%**	31.33%
	ACT	Unconstrained	15.02	9.64	11.00	-37%	12%	**62.54%**
		Constrained	7.79	6.45	11.00	**29%**	**41%**	62.07%
	GCT	Unconstrained	6.66	5.58	12.00	45%	54%	**62.81%**
		Constrained	2.38	1.80	10.00	**76%**	**82%**	62.12%

and spam classification with GPT, while Next Token Accuracy is used to evaluate text generation tasks, such as machine translation with LLaMA on WMT and question answering with DeepSeek on SQuAD.

Encoding. Entropy coding length is adopted to measure the compression level of the model, and three coding schemes are applied, including 0-order EG coding, HM coding, and fixed-length (FL) coding. EG coding is motivated by its compatibility with GGDs with low shape parameters and strong hardware parallelism, which is further applied in practical data encoding. HM coding is used for its near-optimal efficiency and closely approximates entropy. And FL coding only serves as a baseline without entropy coding. All encoding methods use linear quantization with the same step size, with 2^{-8} applied to weights and activations and 2^{-16} to gradients due to their smaller magnitudes.

Hardware. All experiments are conducted on a system equipped with 48 GB NVIDIA L20 GPUs and a 32-core INTEL XEON GOLD 6530 CPU with 754 GB of memory, and all models are trained and inferred using the BF16 format.

D.2 Generalization across Expanded Tasks

We further extend the evaluation of GG Init, ACT, and GCT to a broader range of models and tasks. For text classification, we consider Bart (Lewis et al., 2020) on SST2 (Socher et al., 2013) and GPT-2 (Radford et al., 2019) on Spam (Almeida et al., 2011). For text generation, we evaluate LLaMA-3.2 (Grattafiori et al., 2024) on WMT (Barrault et al., 2019) and Qwen3 (Yang et al., 2025) on HotpotQA (Yang et al., 2018). As reported in Table 7, these methods consistently achieve substantial reductions in the coding rates of weights, activations, and gradients while preserving model performance, in line with the observations in Section 5.4.

Table 8: The system-level metrics for evaluating ACT and GCT across different models and tasks include the per-iteration training time, peak GPU memory consumption, and the communication volume of activations or gradients. ACT with $\lambda_{\text{act}} = 1 \times 10^{-3}$ and GCT with $\lambda_{\text{grad}} = 1 \times 10^{-5}$ are employed as the constrained settings, while standard training with $\lambda_{\text{act}} = 0$ and $\lambda_{\text{grad}} = 0$ serves as the unconstrained baseline for comparison.

Model	Method	Strategy	Training Time	GPU Memory	Communication Volume
BERT-0.1B	ACT	Unconstrained	2 min 45 sec	8.76 GB	0.34 TB
		Constrained	2 min 45 sec	8.76 GB	0.10 TB
	GCT	Unconstrained	2 min 18 sec	6.21 GB	2.95 TB
		Constrained	2 min 32 sec	7.08 GB	1.33 TB
Bart-0.4B	ACT	Unconstrained	4 min 27 sec	10.07 GB	0.05 TB
		Constrained	4 min 35 sec	10.07 GB	0.02 TB
	GCT	Unconstrained	4 min 01 sec	6.44 GB	1.22 TB
		Constrained	6 min 43 sec	7.83 GB	0.93 TB
Qwen-0.6B	ACT	Unconstrained	14 min 12 sec	74.82 GB	0.11 TB
		Constrained	13 min 59 sec	74.82 GB	0.05 TB
	GCT	Unconstrained	12 min 00 sec	62.76 GB	8.11 TB
		Constrained	14 min 45 sec	63.31 GB	3.29 TB
GPT-0.8B	ACT	Unconstrained	31 min 13 sec	36.22 GB	0.86 TB
		Constrained	31 min 16 sec	36.23 GB	0.15 TB
	GCT	Unconstrained	29 min 02 sec	35.79 GB	24.19 TB
		Constrained	30 min 12 sec	38.92 GB	10.37 TB
LLaMA-1B	ACT	Unconstrained	29 min 40 sec	40.82 GB	2.45 TB
		Constrained	29 min 51 sec	40.82 GB	1.04 TB
	GCT	Unconstrained	27 min 17 sec	36.45 GB	118.02 TB
		Constrained	32 min 35 sec	42.41 GB	48.30 TB
Gemma-2B	ACT	Unconstrained	0 min 44 sec	31.76 GB	8.15 GB
		Constrained	0 min 43 sec	31.76 GB	3.71 GB
	GCT	Unconstrained	1 min 00 sec	31.92 GB	0.36 TB
		Constrained	1 min 25 sec	39.12 GB	0.15 TB
Qwen-4B	ACT	Unconstrained	45 min 36 sec	71.81 GB	20.85 GB
		Constrained	45 min 36 sec	71.85 GB	10.81 GB
	GCT	Unconstrained	54 min 00 sec	65.02 GB	17.22 TB
		Constrained	58 min 48 sec	68.38 GB	6.15 TB
DeepSeek-7B	ACT	Unconstrained	40 min 59 sec	79.67 GB	6.38 TB
		Constrained	41 min 05 sec	79.65 GB	2.82 TB
	GCT	Unconstrained	37 min 07 sec	78.92 GB	441.79 TB
		Constrained	45 min 49 sec	87.85 GB	234.17 TB

D.3 GG INIT AND BACKSLASH

GG Init modifies the weight initialization without extra computational cost, requiring only the choice of shape parameters. Using BERT on IMDB, we further investigate GG Init with shapes $\{0.1, 0.5, 1.0, 2.0\}$ interacting with BackSlash training, conventional training serving as a baseline in Table 9. GG Init exhibits strong compatibility with BackSlash, where smaller shape parameters generally yield improvements in both compression and accuracy. While BackSlash exhibited stable compression ratios concerning the initialization shape, accuracy was still notably higher when initialized with GGD using smaller shape parameters. In contrast, models trained conventionally showed high sensitivity to the shape parameter, with compression ratios varying by more than 2.5× between shape parameters 2 and 0.1, verifying the importance of distribution-aware initialization.

Table 9: Performance of GG Init with various shapes using conventional training and BackSlash.

Shape	Method	FL (bits)	EG (bits)	HM (bits)	C_{EG}	C_{HM}	Accuracy
0.1	Conventional	12.00	1.59	1.57	87%	87%	82.23%
	BackSlash	12.00	1.10	1.10	**91%**	**91%**	**83.20%**
0.5	Conventional	10.00	4.97	4.54	50%	55%	76.63%
	BackSlash	10.00	1.99	1.90	**80%**	**81%**	**81.68%**
1.0	Conventional	10.00	6.05	5.12	39%	49%	75.95%
	BackSlash	10.00	2.18	2.06	**78%**	**79%**	**81.87%**
2.0	Conventional	10.00	6.63	5.32	34%	47%	74.12%
	BackSlash	10.00	2.27	2.12	**77%**	**79%**	**78.18%**

D.4 System Metrics

Compared with conventional training, ACT and GCT introduce additional rate regularization terms into the optimization objective, which may incur extra computational and memory overhead. To provide a systematic comparison, we report system-level metrics for all models and tasks during training, including per-epoch training time, per-model peak memory usage, and the corresponding communication volume of activations or gradients, as summarized in Table 8.

Relative to conventional training, ACT does not introduce additional overhead in either training time or memory consumption, while significantly reducing the activation communication across different modules. This indicates that activation-constrained optimization improves both computational and communication efficiency without increasing spatiotemporal complexity.

For GCT, the training time and memory usage exhibit a slight increase. This overhead arises from storing gradients from the previous iteration to accelerate second-order derivative estimation of all weights. Nevertheless, this additional cost is relatively modest and is well justified by the substantial reduction in gradient communication. The advantage of GCT becomes particularly pronounced in bandwidth-constrained scenarios.

E Limitations

Leveraging the statistical properties of the GGDs, we propose GG Init, ACT, and GCT for weights, activations, and gradients for more efficient training and communication, while there are a few limitations currently. On the one hand, although empirically setting the Lagrange multipliers $\lambda_{\text{act}} = 1 \times 10^{-3}$ and $\lambda_{\text{grad}} = 1 \times 10^{-5}$ yields a robust rate–distortion trade-off across different models, these choices are not guaranteed to be optimal; future work will further investigate principled strategies for hyperparameter selection based on rate–distortion theory. On the other hand, the FPGA-based framework for model communication, training, and inference built upon EG coding is still under development, and we believe that the efficient EG coding will be more widely adopted at the hardware implementation in the near future.

F Impact Statement

We introduce a unified LLM optimization framework leveraging generalized Gaussian priors that are inherently compressed and high-performing. Unlike post-training compression, this work optimizes the model's statistical distribution during training to ensure low-entropy representations, thereby facilitating and cooperating with quantization, sparsification, and entropy coding, which will shift attention toward the fundamental question of how the statistical properties of the data influence its own compressibility. Furthermore, our framework substantially reduces the computational and environmental costs of training while accelerating deployment across resource-constrained hardware, federated systems, and energy-efficient AI applications.