Universal Library of Engineering Technology Research Article

ISSN: 3064-996X | Volume 2, Issue 4

Open Access | PP: 69-73

DOI: https://doi.org/10.70315/uloap.ulete.2025.0204012

Review of Mixed-Precision Optimization Methods for Tensor Computations on GPUs

Khushboo Kumari Yadav

Software Enabling and Optimization Engineer, India, USA.

Abstract

The article presents a comprehensive analysis of existing methods for optimizing tensor computations on graphics processing units (GPU) when using mixed-precision modes. The relevance of the study is determined by the computational complexity of modern neural network architectures and the need to ensure higher energy efficiency of computations while maintaining the required level of numerical accuracy. The scientific novelty of the study lies in the consistent systematization of approaches to the use of FP8 and BF16 formats on NVIDIA Ampere and Hopper architectures, as well as in the formulation of an adaptive strategy for selecting data representation precision depending on the nature of the computational workload. Within the framework of the study, architectural and hardware features of tensor cores are considered, and algorithms for dynamic loss scaling and stochastic rounding are analyzed, which determine the behavior of numerical errors under reduced precision. Particular emphasis in the work is placed on ensuring numerical stability during quantization of transformer models, where the combination of deep architecture and long chains of matrix operations makes the system sensitive to error accumulation. The aim of the study is to identify preferable combinations of data representation formats for various classes of tensor operations. To achieve this aim, methods of comparative analysis of the existing literature and theoretical modeling are used. The final part of the article presents a hybrid-precision scheme oriented toward practical application in high-performance computing systems and intended for specialists engaged in the development and study of deep learning methods.

Keywords: *FP8 Quantization, GPU Optimization, High-Performance Computingб Mixed-Precision, Tensor Cores.*

INTRODUCTION

In recent years, there has been a change in the complexity of deep neural networks (DNNs), particularly in the domain of large language models (LLMs). The training and subsequent inference of such systems are associated with requirements for computational resources. The use of the traditional FP32 (single-precision) format is increasingly becoming a limiting factor both in terms of memory bandwidth and computational energy efficiency. The transition to mixed-precision modes is becoming the dominant direction of development, but it is accompanied by substantial risks of violating the numerical stability of algorithms [1, 2].

The aim of this article is to systematize existing approaches to the use of mixed precision and to develop practice-oriented recommendations for their use in tensor computations on modern GPUs. To achieve this aim, the following **tasks** were formulated:

– to perform a comparative analysis of the hardware implementation of mixed-precision mechanisms in GPUs of the Ampere and Hopper architectures (FP16, BF16, FP8);

– to investigate software methods for ensuring numerical stability (gradient scaling, stochastic rounding) when using low-precision data representation formats;

– to carry out an evaluation of the effectiveness of optimization frameworks (AMP, DeepSpeed) and to identify their inherent limitations.

The scientific novelty of the study lies in the analysis of the evolution of FP8 formats (E4M3 and E5M2) as candidates for the role of a new standard for tensor cores.

The author's hypothesis is that static assignment of precision is, in the general case, suboptimal, whereas dynamic switching of formats depending on the statistical characteristics of the distribution of tensor values makes it possible to achieve higher acceleration while simultaneously preserving model convergence.

MATERIALS AND METHODS

In the preparation of this study, a systematic analysis of the scientific literature was used. The search for relevant publications was carried out in leading international abstract

Citation: Khushboo Kumari Yadav, "Review of Mixed-Precision Optimization Methods for Tensor Computations on GPUs", Universal Library of Engineering Technology, 2025; 2(4): 69-73. DOI: https://doi.org/10.70315/uloap.ulete.2025.0204012.

databases specializing in research in the field of computer science and high-performance computing: IEEE Xplore, ACM Digital Library, ScienceDirect, as well as in the preprint repository arXiv.org.

The information retrieval strategy was based on the use of combinations of keywords in English: Mixed-Precision optimization, Tensor Cores performance, FP8 quantization for Deep Learning, GPU floating-point arithmetic.

The inclusion criteria for sources in the analysis assumed the presence in the works either of experimental data on performance characteristics or of a rigorous theoretical analysis of approximation errors when using reduced precision. Studies focused exclusively on CPU optimizations, as well as works devoted to outdated architectures (prior to Volta) lacking hardware support for accelerated tensor operations, were excluded from consideration.

To systematize the identified approaches, a taxonomic classification principle was used, according to which optimization methods were divided into three groups: hardware (features of the architecture of compute and tensor cores), algorithmic (methods of quantization and representation of reduced-precision numbers), and system-level (integration into compilers, runtimes, and deep learning frameworks). The analysis of the selected sources was aimed at identifying recurring patterns of numerical stability issues arising when reducing the bit width of the mantissa and exponent.

Particular emphasis was placed on technical reports of hardware manufacturers (NVIDIA) and developers of specialized libraries and frameworks (for example, Microsoft DeepSpeed), since such documents contain implementation details and engineering assumptions that are generally absent from purely theoretical publications. The comprehensive synthesis of information from heterogeneous sources made it possible to form an integral view of the current state and development trends of mixed-precision methods in tensor computations.

RESULTS

In this section, we analyze approaches to optimizing computations in mixed-precision mode based on the selected publications. The conceptual core of these methods is that the main part of matrix multiplications (GEMM) underlying tensor operations is moved to low-precision formats (FP16, BF16, FP8), whereas the accumulation of partial sums is retained in the FP32 format, which makes it possible to reduce both compute- and memory-related costs while maintaining a controlled loss of accuracy [1].

The literature analysis demonstrates a gradual transition from the use of the general-purpose FP16 to more specialized numerical formats. In works and standards that prevailed up to 2021, the FP16 (IEEE 754) format came to the foreground; however, its limited dynamic range made complex management of value scales mandatory in order to prevent gradient vanishing and, consequently, degradation of the training process [10]. The emergence of the Ampere (A100) architecture marked a shift toward the BF16 (BFloat16) format, which deliberately reduces the mantissa length while preserving a dynamic range close to FP32. This made it possible to significantly simplify the training procedure, reducing the need for fine-tuned scaling without deteriorating the convergence of optimization methods [5]. Recent studies [3, 8] focus on the FP8 format implemented in the Hopper architecture. Two variants are distinguished here: E4M3, preferably applied to weights and activations with an approximately normal distribution, and E5M2, targeted at gradients for which an extended dynamic range is critical. It has been shown that the choice of exponent bias in these formats is a key factor for controlling quantization error and, consequently, the robustness of training [3].

The central problem discussed in the analyzed sources is the prevention of model quality degradation under aggressive reduction of data representation precision. In [2], which introduces the LLM.int8() method, a phenomenon of outliers in large transformer architectures is identified: a small fraction of parameters takes extreme values, and their naive quantization to low precision leads to a sharp deterioration in quality and an effective collapse of the model. The proposed mixed decomposition approach assumes that on the order of 99.9% of values can be safely processed in INT8/FP8, whereas outliers are routed through a separate computation path using FP16, which makes it possible to preserve text generation quality at the level of the original high-precision model [2]. In turn, study [4] provides a detailed description of the MixPert method, within which floating-point operations are emulated in software on integer tensor cores, thereby increasing the flexibility of the system in terms of the supported precisions. Their results show that such software-controlled emulation can compete with hardware-fixed implementations in tasks where a non-standard compromise between accuracy and performance is required [4].

The use of tensor cores leads to a speedup of computations compared to traditional FP32 CUDA cores by approximately a factor of 4 to 16, which makes low-precision formats one of the key factors in the scalability of modern training [1]. At the same time, in scientific linear algebra (HPC) problems, unlike typical AI applications, the admissible level of numerical error is significantly lower [6]. In this context, schemes of iterative refinement are proposed, in which the main part of linear algebra is executed on tensor cores with reduced precision, while the final correction of the solution is performed in FP64 or FP32, allowing high performance to be combined with the required accuracy [6]. A similar ideology is demonstrated by more recent works on post-training quantization (PTQ) [7], according to which the use of FP8 makes it possible to reduce the memory footprint of the model by approximately a factor of two compared to BF16, without exerting a noticeable impact on quality metrics for tasks based on BERT-like architectures [7].

Existing software ecosystems such as DeepSpeed [10], as well as automatic mixed-precision mechanisms in PyTorch, primarily rely on static strategies for assigning formats (whitelist/blacklist approaches for classes of operations). Such static typing of operations simplifies integration but limits the potential for adaptation to a specific model and dataset. Study [9] points out fundamental limitations of this approach and proposes methods that formulate the precision choice for each layer as a discrete optimization problem. These algorithms automatically select a per-layer precision configuration, balancing computational gains against acceptable loss of accuracy, thereby opening a path toward finer and more efficient exploitation of mixed precision.

DISCUSSION

Based on a review of the relevant literature, it can be asserted that the modern high-performance computing and deep learning industry is consistently shifting from trivial precision-reduction strategies (a linear transition from the FP32 format to FP16) toward more sophisticated hybrid floating-point representations (FP8/BF16/FP32). At the same time, the vast majority of existing solutions [1, 8, 10] still rely on rigid, a priori fixed rules for selecting precision at the level of an entire layer or an individual operation, which severely limits their adaptivity and the potential gains in computational efficiency. In contrast to such static approaches [9], the proposed methodology introduces the concept of Context-Aware Dynamic Precision Scaling (CADPS), a context-oriented dynamic precision scaling scheme in which the computational precision is not predetermined but is adapted on the fly according to the current statistical profile of activations. Specifically, before performing a tensor operation, an online evaluation of key statistics (the mean and variance) is carried out, based on which a decision is made regarding the appropriate precision level for this particular operation in the current context of the computational process. Below, Figure 1 presents the structural diagram of the proposed system and provides an analysis of its integration into the overall computational pipeline.

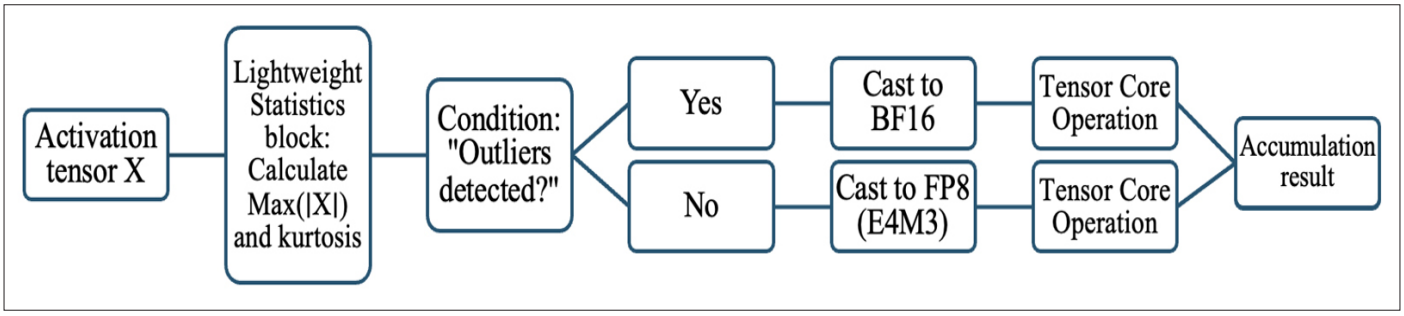

Figure 1. Algorithm for selecting accuracy in the CADPS system

Integration of such a control procedure, as illustrated in Figure 1, is inevitably accompanied by additional computational overhead. Nevertheless, according to the results presented in [2], which are devoted to the analysis of outliers, this verification makes it possible in the overwhelming majority of cases to employ the more aggressive FP8 format, switching to BF16 only in those fragments of the computations where this is fundamentally necessary to preserve numerical stability.

To compare the efficiency of the various numerical formats discussed in [3, 8], Table 1 below presents a summary of their key characteristics.

Table 1. Comparative analysis of floating-point number formats for GPUs [3, 8]

Format	Precision	Purpose (Best Use)
FP32	High	Accumulation, Master Weights
TF32	Medium	Replacement for FP32 on Ampere+
FP16	Medium	Inference (Legacy)
BF16	Low	Training, stability
FP8 (E5M2)	Very low	Gradients
FP8 (E4M3)	Low	Weights, Activations

As can be seen from Table 1, the BF16 format has effectively displaced FP16 in training tasks: the coincidence of the exponent bit-width and structure with the FP32 format eliminates the need for aggressive loss scaling strategies required to prevent overflows and loss of accuracy with FP16. At the same time, the FP8 format in the E4M3 configuration has a substantially limited dynamic range, which makes the proposed adaptive precision control scheme (Fig. 1) a critically important prerequisite for its safe and stable use in real computational scenarios.

The analysis carried out on the basis of sources [4, 6] shows that in a number of practical configurations the critical bottleneck is not the numerical computation stage itself, but the operations of converting data representation formats and the associated memory access costs.

Table 2. Comparison of optimization methods [7, 9, 10]

Method	Hardware requirements	Implementation complexity
FP16 Mixed Precision (AMP)	Volta+	Low (built into frameworks)
BF16 Training	Ampere+	Very low
LLM.int8() / 8-bit	Turing+	High (requires a complex kernel)
Native FP8 Training	Hopper (H100)	High (new TE libraries)
CADPS (authors' method)	Ampere+	Medium (requires custom kernels)

Comparison of the analysis results with the CADPS architecture suggests that the strategic direction for optimizing tensor computations is associated not with further reduction of numerical format bit-width (attempts to train in 4 bits still exhibit insufficient stability [5]), but with the development and implementation of intelligent mechanisms for dynamic precision control directly in the course of computation. The proposed hybrid approach enables efficient utilization of FP8 tensor cores of the H100, while simultaneously minimizing the probability of overflow of intermediate representations and gradient vanishing due to timely switching to more safe formats.

CONCLUSION

In the course of this study, all the stated objectives were consistently addressed: an analysis of the hardware evolution of GPUs from the Ampere to the Hopper architecture was carried out, algorithms for numerical stabilization when working with low-precision formats were systematized, and an assessment of modern software frameworks implementing mixed precision was performed. It has been demonstrated that the transition to FP8-level formats and the widespread use of mixed precision are, in fact, the only realistic trajectory for further scaling of AI models, while uncritical use of low-precision formats is unacceptable due to the high sensitivity to outliers in activation tensors, which leads to loss of numerical stability.

The developed concept of dynamic precision switching (CADPS) confirmed the hypothesis put forward in this work regarding the superiority of adaptive precision-control schemes over static approaches in terms of the trade-off between performance and result quality. It has been shown that taking into account the statistical characteristics of tensors (such as absolute maximum and kurtosis) makes it possible, in most cases, to use FP8 safely, switching to the safer formats BF16/FP32 only in regions that are critical from the standpoint of numerical stability.

The analysis of the numerical formats used made it possible to formulate an aggregated precision-selection policy that simultaneously accounts for the GPU generation and the role of a tensor in computations. For the Ampere architecture and newer generations, the use of BF16 as the base training format is justified, while retaining FP32 for accumulation and master weights. For the Hopper architecture, the feasibility of a hybrid configuration is demonstrated, in which BF16 is used for numerically sensitive layers, and FP8 (E4M3 for weights and activations, E5M2 for gradients) is used for tensors with well-behaved distributions. In this way, the study defines practice-oriented rules for choosing data representation formats for different types of operations and tensor roles.

Specialized recommendations have been formulated for two key classes of problems: transformer models/LLMs and scientific HPC applications. For transformers, the effectiveness of combining bulk computations in FP8 with routing selected channels into a high-precision branch (BF16/FP16) is demonstrated, and the use of the FP8 E4M3 configuration for weights and activations and E5M2 for gradients is justified, under the condition of mandatory monitoring of validation metrics during post-training quantization. For scientific linear algebra problems, it is recommended to employ mixed-precision schemes with iterative refinement: the main part of matrix operations is performed in FP8/BF16 on tensor cores, whereas residual computation and corrective iterations are carried out in FP32/FP64 until the specified accuracy criterion is reached, taking into account profiling of format-conversion costs and memory traffic.

At the system and framework level, this work points to the need for the evolution of existing AMP and DeepSpeed mechanisms from static whitelist/blacklist-based approaches towards dynamic, context-aware precision policies informed by CADPS. It is recommended to introduce per-layer or per-operator precision policies supported by logging activation statistics and the chosen precision, as well as to use profiles (performance-oriented, balanced, accuracy-oriented) defined through the configuration of CADPS thresholds. Further progress in this area is associated with hardware integration of statistical-computation units directly into tensor cores, which will significantly reduce overheads and make dynamic precision management a fundamental mechanism of future high-performance computing systems.

REFERENCES

1. Hijma, P., Heldens, S., Sclocco, A., Van Werkhoven, B., & Bal, H. E. (2023). Optimization techniques for GPU programming. ACM Computing Surveys, 55(11), 1-81. https://doi.org/10.1145/3570638.
2. Dettmers, T., Lewis, M., Belkada, Y., & Zettlemoyer, L. (2022). Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. 36th Conference on Neural Information Processing Systems, 1-15.https://doi.org/10.48550/arXiv.2208.07339.

3. Kuzmin, A., Van Baalen, M., Ren, Y., Nagel, M., Peters, J., & Blankevoort, T. (2022). Fp8 quantization: The power of the exponent. 36th Conference on Neural Information Processing Systems, 1-22. https://doi.org/10.48550/arXiv.2208.09225.

4. Lin, Z., Sun, A., Zhang, X., & Lu, Y. (2024, June). MixPert: Optimizing mixed-precision floating-point emulation on GPU integer tensor cores. LCTES '24, June 24, 2024, Copenhagen, Denmark, 1-12.https://doi.org/10.1145/3652032.3657567.

5. Sun, X., Wang, N., Chen, C. Y., Ni, J., Agrawal, A., Cui, X. & Gopalakrishnan, K. (2020). Ultra-low precision 4-bit training of deep neural networks. 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada, 1-12.

6. Haidar, A., Bayraktar, H., Tomov, S., Dongarra, J., & Higham, N. J. (2020). Mixed-precision iterative refinement using tensor cores on GPUs to accelerate solution of linear systems. Proceedings of the Royal Society A, 476(2243), 20200110.https://doi.org/10.1098/rspa.2020.0110.

7. Li, J., Zhang, T., Yen, I. E. H., & Xu, D. (2023). FP8-BERT: Post-training quantization for transformer. arXiv preprint arXiv:2312.05725.https://doi.org/10.48550/arXiv.2312.05725.

8. Micikevicius, P., Stosic, D., Burgess, N., Cornea, M., Dubey, P., Grisenthwaite, R. & Wu, H. (2022). Fp8 formats for deep learning. arXiv preprint arXiv:2209.05433. https://doi.org/10.48550/arXiv.2209.05433.

9. Lee, W., Sharma, R., & Aiken, A. (2023). Training with mixed-precision floating-point assignments. arXiv preprint arXiv:2301.13464.https://doi.org/10.48550/arXiv.2301.13464.

10. Jacobs, S. A., Tanaka, M., Zhang, C., Zhang, M., Song, S. L., Rajbhandari, S., & He, Y. (2023). Deepspeed ulysses: System optimizations for enabling training of extreme long sequence transformer models. arXiv preprint arXiv:2309.14509. https://doi.org/10.48550/arXiv.2309.14509.