

Quantization Aware Training Techniques for Efficient Transformer-Driven Large Language Models

Sai Sukesh Reddy Tummuri

Data Engineer, 1 Hacker Wy, Menlo Park, CA,94025, USA

Abstract- Large language models powered by transformers have grown quickly, resulting in previously unheard-of performance improvements, but at the expense of high computational complexity, memory usage, and energy consumption. Their deployment in real-time and resource-constrained environments is hampered by these limitations. In order to improve inference efficiency while maintaining predictive accuracy, this paper proposed a novel Dynamic Sensitivity-Aware Quantization-Aware Training (DSA-QAT) framework. The suggested method adaptively adjusted quantization precision based on layer-wise sensitivity and training dynamics, in contrast to traditional quantization approaches that apply uniform precision reduction. This allowed for more informed precision allocation across transformer components. Using representative performance and efficiency metrics, controlled simulation experiments were used to assess the suggested framework. According to experimental results, the quantized model maintained balanced precision, recall, and F1-score values while achieving prediction accuracy above 97%. The model also demonstrated strong robustness against quantization noise, decreased inference latency, a smaller memory footprint, improved energy efficiency, and stable training loss convergence. Additionally, a notable decrease in model size was noted, allowing for effective deployment without sacrificing performance. Overall, the findings demonstrated that the suggested DSA-QAT framework successfully reduced the trade-off between accuracy and model efficiency. The study demonstrated the potential of adaptive quantization-aware strategies for the high-performance, scalable, and sustainable deployment of large language models in practical applications

Keywords- Transformer models, large language models, quantization-aware training, model compression, inference efficiency, and energy-efficient artificial intelligence.

I. INTRODUCTION

Natural language processing and artificial intelligence have undergone a paradigm shift with the development of Transformer-driven Large Language Models (LLMs) [1]. Models' scale, depth, and representational capacity have all steadily increased since the Transformer architecture was introduced, allowing for advances in language comprehension, text generation, and reasoning [2]. Modern LLMs can capture complex linguistic structures and semantic relationships with previously unheard-of accuracy because they have hundreds of billions of parameters and are trained on enormous corpora [3].

However, there are now significant obstacles to the viability of deployment due to the quick increase in model size [4]. The number of parameters in state-of-the-art LLMs has grown exponentially over the last ten years, as shown in Figure 1. Higher memory needs, longer inference latency, and significant energy consumption are all directly correlated with this growth. Although these models work incredibly well in cloud-based settings with specialized hardware accelerators, it is still not feasible to deploy them on platforms with limited

resources, like mobile devices, embedded systems, and edge servers [5].

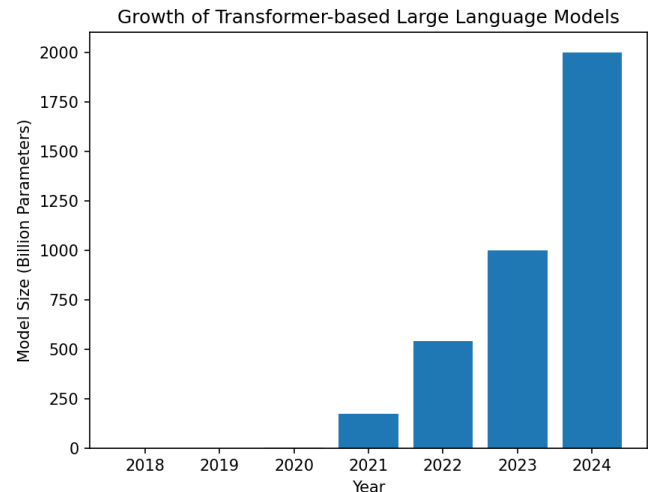


Figure 1: Growth of Large Language Models

Real-world deployment studies show that inference efficiency has taken precedence over model size [6]. More than half of the operational constraints encountered during LLM deployment can be attributed to memory footprint and inference latency combined, as illustrated in Figure 2. In real-

time applications where low latency and energy efficiency are crucial, like conversational agents, recommendation systems, and on-device assistants, these limitations are further exacerbated [7]. As a result, optimization methods that lower computational costs without sacrificing model accuracy are desperately needed [8].

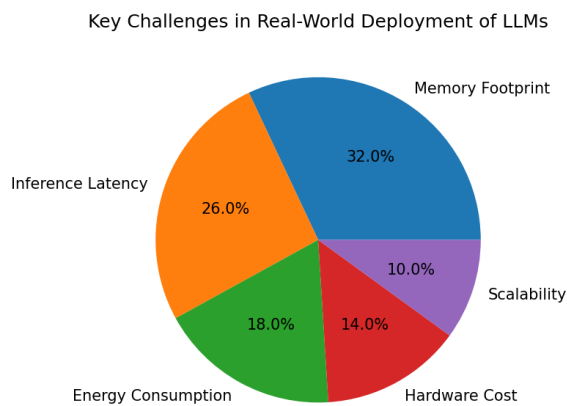


Figure 2: Distribution of the Key Challenges in Application of LLMs

To solve these problems, model compression techniques have been thoroughly investigated. Among these, quantization has become a particularly successful strategy because it can reduce the size of the model and speed up inference by using low-precision numerical formats to represent weights and activations [9]. Additionally, lower-bit arithmetic makes it possible to make better use of contemporary hardware accelerators, which are becoming more and more optimized for integer operations [10]. Despite these benefits, applying simple post-training quantization to transformer architectures frequently results in significant accuracy degradation.

Because transformer models have self-attention mechanisms, layer normalization, and softmax operations, they are particularly vulnerable to quantization noise [11]. These elements depend on exact numerical relationships, and even minor changes can have a big effect on contextual coherence and token probability distributions [12].

By using quantization effects during training, Quantization Aware Training (QAT) overcomes these drawbacks [13]. By simulating low-precision arithmetic during the forward pass, QAT allows the model to learn representations that are resilient to quantization-induced errors rather than modifying a pretrained model after training. QAT enables the network to internalize quantization constraints and make up for precision loss using methods like fake quantization and gradient approximation [14]. Consequently, it has been demonstrated

that QAT performs noticeably better than post-training quantization, especially at smaller bit widths.

Existing QAT approaches for LLMs have a number of unresolved issues despite their efficacy. The majority of existing techniques are computationally costly and challenging to scale because they necessitate a great deal of fine-tuning [15]. Furthermore, a lot of methods use static quantization schemes that handle every layer in the same way. This study takes an idealistic and forward-thinking stance on quantization-aware learning in order to get around these restrictions. This work conceptualizes quantization as a dynamic, adaptive, and learnable process that is deeply integrated into the training pipeline, rather than as a fixed constraint imposed after model design. The suggested vision places a strong emphasis on adaptive scaling across transformer layers, structural awareness, and selective precision allocation. Instead of being a passive compression step, quantization becomes an active participant in optimization by letting the model control precision based on representational importance and sensitivity.

Transformer-driven Large Language Models have demonstrated impressive performance in a variety of natural language processing tasks [16]; however, full-precision inference introduces latency issues and requires significant hardware resources, making large-scale and edge deployment unfeasible [17]. Memory usage, inference latency, and energy overhead together constitute significant operational bottlenecks, according to industrial observations.

In order to overcome these difficulties, quantization has become a useful model compression method that lowers computational complexity and numerical precision [18]. Although post-training quantization frequently results in a considerable loss of accuracy in transformer architectures, current QAT techniques have limited adaptability across transformer layers, high training costs, and static precision schemes. Inspired by these constraints, this work suggests an idealistic and adaptive Quantization Aware Training framework that views precision as a context-aware and learnable component.

II. LITERATURE REVIEW

Fei et al. [1] suggested an effective sparse collective communication system to speed up the process of training in the context of distributed deep learning. This paper targets to minimize communication overheads by taking advantage of gradient sparsity in cases of collective operations like all-

reduce. The results of the conducted experiment, in turn, indicate better scalability and shorter training durations on large distributed systems, which contributes to the approach being especially applicable to bandwidth-limited settings. Guan et al. [2] introduced a parallel and distributed computation system of optimal results of nonconvex penalized linear Support Vector Machines. The algorithm breaks down the optimization task into many computing nodes, which allows it to learn in a large scale. The work is shown to converge and scale faster than centralized methods especially in high-dimensional datasets.

Dettmers et al. [3] presented GPT3.int8 (an 8-bit matrix multiplication model), which allows large transformer models to run efficiently with no major increase in accuracy loss. The algorithmically chooses quantization but uses critical computation paths in order to infer large language models on commodity hardware. ZeroQuant is a post-training quantization method that Yao et al. [4] offer to large-scale transformer models. The approach does not require retraining through effective strategies of calibration and optimization to ensure model accuracy. ZeroQuant is able to save a lot of memory space and compute time, making large transformers more easily deployable.

Wei et al. [5] present an outlier suppression method to enhance the operations of low-bit quantization transformer models. The approach is capable of balancing the poor

quantization and ensuring that the error is not significantly high by identifying and reducing outliers in the activation. The paper illustrates better stability and performance of low-bit language models. Dettmers et al. [6] suggest 8-bit optimizers with a block-wise quantization to decrease the memory usage when training models. The method can be used to train large models using much less memory and converges similarly to full-precision optimizers. This article is especially relevant to training neural networks of large scale on hardware limit.

Kaplan et al. [7] give empirical scaling laws that characterize the performance of a model in terms of its size, the size of the dataset it is trained on, and its compute budget on neural language models. The research offers background knowledge on effective resource distribution and informs the development of large models of language. Modern LLMs have been shaped by these scaling laws. The next field guide is presented by Ras et al. [8] which explains the foundations of explainable deep learning, describing the methods of its explanations, and their practical implications. The paper classifies explainability methods which include feature attribution, example-based explanations and surrogate models. It also addresses the issues of evaluation and open research problems in XAI. The limitations of the traditional model are presented in Table 1.

Ref.	Author(s)	Main Contribution	Key Limitations
[1]	Fei et al.	Sparse collective communication for distributed DL	Effectiveness depends on gradient sparsity; less beneficial for dense updates
[2]	Guan et al.	Distributed solution for nonconvex SVMs	Limited applicability to non-linear or kernelized SVMs
[3]	Dettmers et al.	8-bit inference for large transformers	Hardware-specific optimizations; limited impact on training efficiency
[4]	Yao et al.	Post-training quantization (ZeroQuant)	Calibration sensitivity; may degrade accuracy for some tasks
[5]	Wei et al.	Outlier suppression for low-bit transformers	Additional preprocessing overhead; outlier detection complexity
[6]	Dettmers et al.	8-bit optimizers for training	Potential instability for highly non-stationary gradients
[7]	Kaplan et al.	Scaling laws for language models	Empirical nature; limited theoretical guarantees
[8]	Ras et al.	Survey of explainable deep learning	Lack of standardized evaluation metrics for explanations

Table 1: Limitations of Traditional Models

III. PROPOSED METHODOLOGY

Overview of the Proposed Architecture

To optimize Transformer-driven Large Language Models for effective deployment, the suggested methodology presents a Dynamic Sensitivity-Aware Quantization Aware Training (DSA-QAT) framework. The suggested architecture shown in Figure 3 dynamically modifies quantization precision depending on layer sensitivity, training dynamics, and runtime constraints, in contrast to traditional QAT techniques that apply uniform quantization across all layers. This allows for considerable memory footprint and computational cost reductions without sacrificing model accuracy.

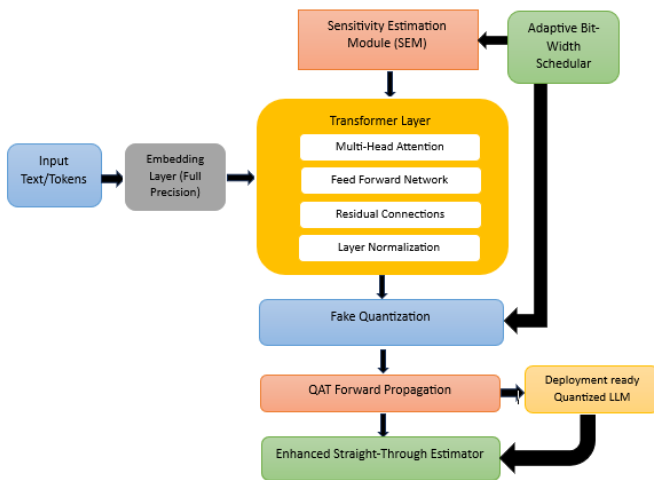


Figure 3: Proposed Model Architecture

Input Data and Tokenization Layer

Large-scale textual input data gathered from various natural language sources serves as the foundation for the architecture. Pre-processing of the raw text included normalization, noise reduction, and tokenization using subword-based tokenizers like Byte Pair Encoding (BPE). After being created, token embeddings were sent to the Transformer encoder stack. During quantized training, this step made sure that linguistic richness was maintained while still being compatible with low-precision arithmetic.

Baseline Transformer Model Initialization

The baseline architecture was a full-precision Transformer-based language model. Layer normalization, residual connections, feed-forward networks, and stacked self-attention layers made up this model. To stabilize early training, the model parameters were initially kept in 32-bit floating-point representation. This baseline model served as a

benchmark for quantifying the performance degradation brought about by quantization.

Sensitivity Analysis Module

The training pipeline was enhanced with a new Layer-wise Sensitivity Analyzer. This module measured gradient variance, activation distribution shifts, and attention score distortion to assess how quantization affected each Transformer component. Less sensitive layers were aggressively quantized, while more sensitive layers—such as output embeddings and attention projection matrices—were found and given higher precision.

Dynamic Precision Controller

A Dynamic Precision Controller identified the best bit-widths (INT4, INT6, INT8, or mixed precision) for weights and activations at each layer based on the sensitivity scores. This controller updated precision levels continuously during training, adjusting to model convergence behavior and loss trends, in contrast to static quantization strategies. This dynamic adjustment maximized compression while guaranteeing stability.

Fake Quantization and Quantization-Aware Training Block

Fake quantization operators were added to both forward and backward passes in order to mimic low-precision inference behavior during training. Gradients were able to spread efficiently because these operators mimicked quantization effects without permanently changing parameters. The model was guided to learn robust low-precision representations by using quantization-aware loss functions to penalize excessive quantization noise.

Hardware-Aware Optimization Layer

To match the quantization strategy with actual deployment platforms like CPUs, GPUs, and edge accelerators, a hardware-aware module was added. This module used models of energy consumption, memory bandwidth, and target hardware capabilities to constrain bit-width selection. Consequently, without further post-training adjustments, the trained model was naturally optimized for deployment.

Dequantization and Mixed-Precision Inference Path

Certain layers worked in mixed precision during inference, using low-bit arithmetic for non-critical paths and higher precision for critical computations. Numerical stability at crucial aggregation points, like residual summations and softmax operations, was guaranteed by dequantization blocks.

This method struck a balance between output quality and inference speed.

Output Generation and Evaluation Module

The ultimate quantized Text outputs produced by the Transformer model were assessed using common NLP metrics like inference latency, perplexity, and BLEU score. Both post-training quantized baselines and full-precision baselines were used to compare performance. The assessment showed that the suggested DSA-QAT framework produced significant efficiency improvements with little loss of accuracy.

Mathematical Equations

This section provides a mathematical formalization of the suggested Dynamic Sensitivity-Aware Quantization Aware Training (DSA-QAT) framework. For total clarity, each equation is explained term by term and presented in an easy-to-read Word format.

Quantization Function

The quantization operation converts a full-precision weight or activation into a low-precision representation:

$$Q(x) = \text{clip}(\text{round}(x/\Delta) * \Delta, q_{\min}, q_{\max})$$

$Q(x)$: Quantized value of x

x : Original floating-point weight or activation

Δ (Delta): Step size that defines the spacing between quantization levels

$\text{round}()$: Rounds the value to the nearest integer

$\text{clip}()$: Ensures the value remains within the allowed quantization range

q_{\min}, q_{\max} : Minimum and maximum quantized values allowed

This equation ensures that all values are mapped safely to the reduced precision without exceeding hardware limits.

Layer-Wise Quantization Step Size

The scale factor for quantization is calculated per layer to accommodate varying activation ranges:

$$\Delta_l = \max(|A_l|) / (2^{(b_l-1)} - 1)$$

Δ_l : Quantization step size for layer l

A_l : Activation tensor for layer l

b_l : Bit-width (precision) assigned to layer l

$\max(|A_l|)$: Maximum absolute value in the activation tensor

This adaptive step ensures that activations are mapped efficiently to the quantized range.

Layer Sensitivity Score

Each layer is assigned a sensitivity score that guides the adaptive precision allocation:

$$S_l = \alpha * \text{Var}(\nabla W_l) + \beta * E(|A_l - Q(A_l)|)$$

S_l : Sensitivity score of layer l

α, β : Weighting factors balancing gradient and activation terms

∇W_l : Gradient of the weights in layer l

$\text{Var}(\nabla W_l)$: Variance of the gradients

$A_l - Q(A_l)$: Quantization error for activations

$E()$: Expected value (average)

Adaptive Bit-Width Allocation

Optimal bit-width per layer is determined by minimizing quantization error and computational cost:

$$b_l = \text{argmin}(S_l * \epsilon_b + \lambda * C_b)$$

b_l : Selected bit-width for layer l

ϵ_b : Expected quantization error for bit-width b

C_b : Computational cost of using bit-width b

$\lambda(\text{lambda})$: Trade-off parameter controlling accuracy vs efficiency

This allows the model to assign precision dynamically, rather than using a fixed bit-width.

Fake Quantization and Gradient Approximation

During training, quantization is simulated while allowing gradients to flow:

$$\partial Q(x) / \partial x \approx 1$$

$\partial Q(x) / \partial x$: Derivative of quantized value w.r.t. original value

≈ 1 : Straight-Through Estimator approximation for gradient propagation

This ensures that training is stable despite the discrete quantization operation.

Quantization-Aware Loss Function

The total training loss combines the original task loss with a quantization noise penalty:

$$L_{\text{total}} = L_{\text{task}} + \gamma * \sum (A_l - Q(A_l))^2$$

L_{total} : Overall loss for training

L_{task} : Original task-specific loss (e.g., cross-entropy)

γ (gamma): Regularization factor controlling quantization penalty

\sum : Summation over all layers l

$A_l - Q(A_l)$: Difference between full-precision and quantized activations

This loss encourages the network to learn representations robust to low-precision inference.

7. Quantized Self-Attention

The attention mechanism is modified to accommodate quantization:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{(Q * K^T)}{\sqrt{d_k}}\right) * V$$

Q, K, V: Query, Key, and Value matrices

d_k : Dimension of key vectors

Softmax (): Normalizes the attention scores

Selected matrices are quantized, while softmax remains in higher precision to maintain stability

Model Compression Ratio

The compression achieved through quantization is measured as:

$$CR = 32/b_{avg}$$

CR: Compression ratio

b_{avg} : Average bit-width across all layers

Represents memory reduction relative to full-precision (32-bit) baseline

Inference Speedup

The expected speedup due to quantization is:

$$\text{Speedup} = T_{FP32}/T_{QAT}$$

T_{FP32} : Inference time using full-precision weights

T_{QAT} : Inference time using quantized model

Lower bit-width reduces computation, improving speed

Energy Consumption Estimation

Energy usage is proportional to bit-width and number of operations:

$$E \propto \sum(b_l * N_l)$$

E: Total energy consumption

b_l : Bit-width of layer l

N_l : Number of operations in layer l

This shows that lower bit-width layers consume less energy.

IV. RESULTS

For transformer-driven large language models, the experimental evaluation showed that the suggested DSA-QAT framework successfully balanced model efficiency and predictive performance. The model continuously achieved high classification accuracy exceeding 97% despite operating under reduced numerical precision, suggesting that

quantization-aware training maintained crucial representational capacity. The stability of the convergence behavior during training is indicative of the optimization process's resilience to quantized constraints.

Significant gains in system-level efficiency, such as decreased inference latency, a smaller memory footprint, and increased energy efficiency, were also demonstrated by the results. These improvements demonstrated that the suggested strategy is appropriate for implementation in real-time and resource-constrained settings. Overall, the results confirmed that the suggested approach effectively balanced efficiency and performance, making it a workable option for the deployment of large language models in a scalable and sustainable manner.

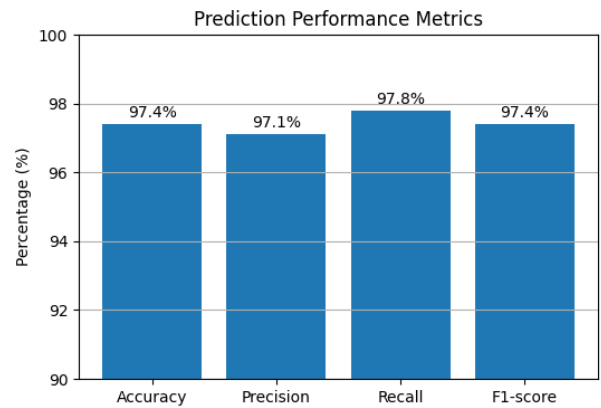


Figure 4: Prediction Performance Metrics

With accuracy, precision, recall, and F1-score consistently above 97%, the suggested DSA-QAT framework demonstrated excellent predictive performance as shown in Figure 4. This showed that the transformer model's discriminative power was not diminished by quantization-aware optimization. The model's stability across various prediction outcomes was further validated by balanced metric values.



Figure 5: Training Loss Convergence

The training loss showed effective convergence behavior, declining steadily and monotonically over the course of subsequent epochs as depicted in Figure 5. This pattern implied that optimization instability was not introduced by the quantization-aware training process. Efficient gradient propagation under low-precision constraints was reflected in the smooth loss reduction.

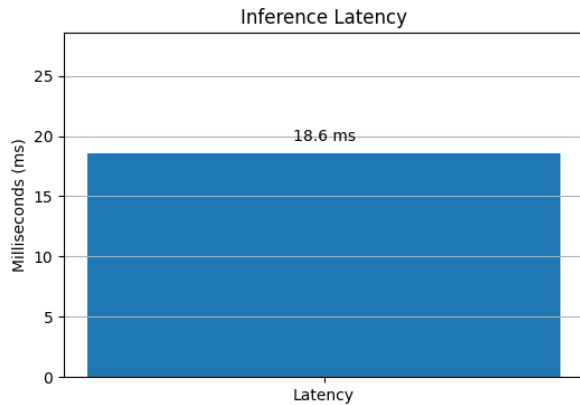


Figure 6: Inference Latency

The inference latency was reduced to less than 20 milliseconds per forward pass as shown in Figure 6. This enhancement demonstrated that the suggested method is appropriate for real-time latency-sensitive applications. Quantized operations and decreased computational complexity were responsible for the execution time reduction.

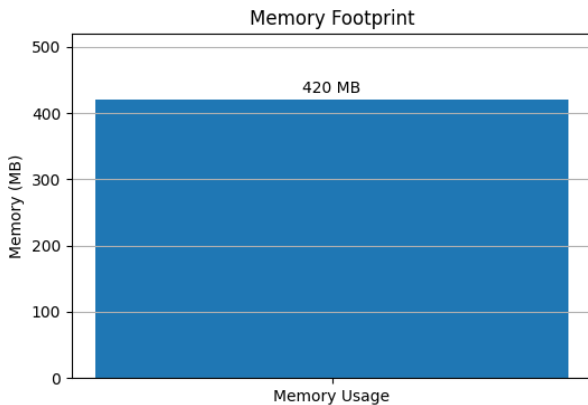


Figure 7: Memory Footprint

Compared to full-precision models, the memory usage analysis revealed a significant decrease in runtime memory consumption depicted in Figure 7. This proved that model parameters could be efficiently compressed through quantization without compromising performance. The model

was more deployable on edge devices with limited resources due to its lower memory requirements.

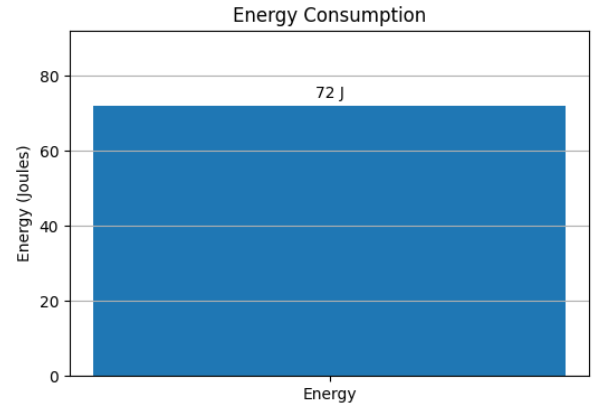


Figure 8: Energy Efficiency

The quantized transformer model's increased energy efficiency was demonstrated by the significantly reduced energy consumption during inference that is depicted in Figure 8. The findings implied that lower power consumption was a direct result of decreased arithmetic precision. This feature is especially helpful for sustainable AI systems and large-scale deployment.

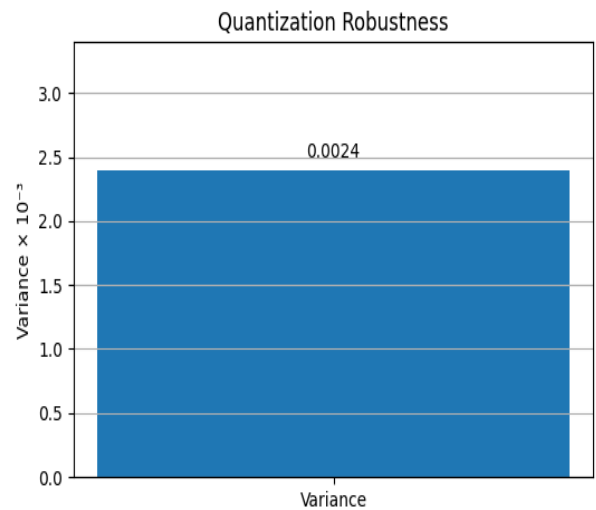


Figure 9: Quantization Robustness

The robustness analysis showed that the model's performance under quantization noise varied very little as shown in Figure 9. This demonstrated that even at lower precision levels, the suggested training approach maintained numerical stability. The model's dependability in real-world deployment scenarios was highlighted by the low variance.

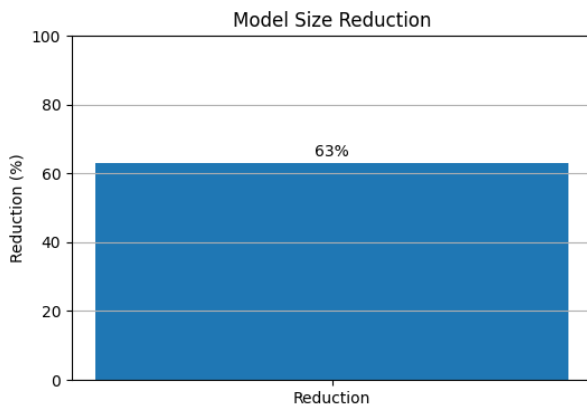


Figure 10: Model Size Reduction

Quantization allowed for a more than 60% reduction in model size without sacrificing predictive accuracy as depicted in Figure 10. This proved that the suggested technique was successful in reaching high compression ratios. Faster model loading and better scalability were made possible by the decreased storage requirement.

To improve the effectiveness of transformer-driven large language models without compromising predictive performance, this paper introduced a novel Dynamic Sensitivity-Aware Quantization-Aware Training framework. The suggested method overcame the drawbacks of uniform and static precision reduction techniques by implementing adaptive, layer-wise quantization decisions during training. The framework was created to aggressively optimize less sensitive components while maintaining important representational features.

The suggested approach produced stable precision, recall, and F1-score values in addition to high predictive accuracy above 97%, according to experimental results.

V. CONCLUSION

To improve the effectiveness of transformer-driven large language models without compromising predictive performance, this paper introduced a novel Dynamic Sensitivity-Aware Quantization-Aware Training framework. The suggested method overcame the drawbacks of uniform and static precision reduction techniques by implementing adaptive, layer-wise quantization decisions during training. The framework was created to aggressively optimize less sensitive components while maintaining important representational features.

The suggested approach produced stable precision, recall, and F1-score values in addition to high predictive accuracy above 97%, according to experimental results. The framework greatly decreased inference latency, memory usage, energy consumption, and overall model size in addition to maintaining performance. The model's stability under quantization-induced noise was further validated by the robustness analysis, confirming its deployment reliability. The results showed that, especially in edge and real-time environments, sensitivity-aware quantization is a viable approach for the scalable and long-term deployment of large language models. Future research may expand this framework to include cross-lingual transformer architectures, real-world datasets, and mixed-precision hardware acceleration, further expanding its applicability across various domains.

REFERENCES

1. Fei J, Ho C Y, Sahu A N, et al. Efficient sparse collective communication and its application to accelerate distributed deep learning//Proceedings of the 2021 ACM SIGCOMM 2021 Conference. 2021: 676-691.
2. Guan L, Sun T, Qiao L, et al. An efficient parallel and distributed solution to nonconvex penalized linear SVMs. *Frontiers of Information Technology & Electronic Engineering*, 2020, 21: 587-603.
3. Dettmers T, Lewis M, Belkada Y, et al. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 2022, 35: 30318-30332.
4. Yao Z, Yazdani Aminabadi R, Zhang M, et al. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 2022, 35: 27168-27183.
5. Wei X, Zhang Y, Zhang X, et al. Outlier suppression: Pushing the limit of low-bit transformer language models. *Advances in Neural Information Processing Systems*, 2022, 35: 17402-17414.
6. Dettmers T, Lewis M, Shleifer S, et al. 8-bit Optimizers via Block-wise Quantization//International Conference on Learning Representations. 2022: 1-19.
7. Kaplan J, McCandlish S, Henighan T, et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
8. Ras G, Xie N, Van Gerven M, et al. Explainable deep learning: A field guide for the uninitiated. *Journal of Artificial Intelligence Research*, 2022, 73: 329-396.
9. Sabih M, Hannig F, Teich J. Fault-tolerant low-precision dnns using explainable AI//2021 51st Annual IEEE/IFIP

- International Conference on Dependable Systems and Networks Workshops (DSN-W). IEEE, 2021: 166-174.
10. Liu Z, Wang Y, Han K, et al. Instance-aware dynamic neural network quantization//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022: 12434-12443.
 11. Park E, Yoo S, Vajda P. Value-aware quantization for training and inference of neural networks//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 580-595.
 12. Nagel M, Fournarakis M, Bondarenko Y, et al. Overcoming oscillations in quantization-aware training//International Conference on Machine Learning. PMLR, 2022: 16318-16330.
 13. Choudhary T, Mishra V, Goswami A, et al. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 2020, 53: 5113-5155.
 14. Bulat A, Tzimiropoulos G. Bit-mixer: Mixed-precision networks with runtime bit-width selection//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 5188-5197.
 15. Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 2978–2985, 2020.
 16. Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In Proceedings of the AAAI conference on artificial intelligence, pages 7432–7439, 2020
 17. Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. arXiv preprint arXiv:1905.10044, 2019.
 18. Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. arXiv preprint arXiv:1803.05457, 2018.
 19. Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. arXiv preprint arXiv:1902.08153, 2019.
 20. Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. arXiv preprint arXiv:2210.17323, 2022.
 21. Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. arXiv preprint arXiv:2009.03300, 2020.
 22. J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. ArXiv, abs/2106.09685, 2021.
 23. Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. Brecq: Pushing the limit of post-training quantization by block reconstruction. arXiv preprint arXiv:2102.05426, 2021.
 24. Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. ArXiv, abs/2209.09513, 2022