EdgeQAT: Entropy and Distribution Guided Quantization-Aware Training for the Acceleration of Lightweight LLMs on the Edge

Anonymous ACL submission

Abstract

Despite the remarkable strides of Large Language Models (LLMs) in various fields, the wide applications of LLMs on edge devices are limited due to their massive parameters and computations. To address this, quantization is commonly adopted to generate lightweight LLMs with efficient computations and fast inference. However, Post-Training Quantization (PTQ) methods dramatically degrade in quality when quantizing weights, activations, and KV cache together to below 8 bits. Besides, many Quantization-Aware Training (QAT) works quantize model weights, leaving the activations untouched, which do not fully exploit the potential of quantization for inference acceleration on the edge. In this paper, we propose EdgeQAT, the Entropy and Distribution Guided QAT for the optimization of lightweight LLMs to achieve inference acceleration on Edge devices. We first identify that the performance drop of quantization primarily stems from the information distortion in quantized attention maps, demonstrated by the different distributions in quantized query and key of the self-attention mechanism. Then, the entropy and distribution guided QAT is proposed to mitigate the information distortion. Moreover, we design a token importance-aware adaptive method to dynamically quantize the tokens with different bit widths for further optimization and acceleration. Our extensive experiments verify the substantial improvements with our framework across various datasets. Furthermore, we achieve an on-device speedup of up to 2.37× compared with its FP16 counterparts across multiple edge devices, signaling a groundbreaking advancement.

1 Introduction

Large Language Models (LLMs) (Zhang et al., 2022; Radford et al., 2019b; Brown et al., 2020a,b; Touvron et al., 2023) based on Transformers (Vaswani et al., 2017) have emerged as the dominant force in the field of Natural Language Processing (NLP). There is a growing trend of integrating LLMs for various applications to optimize user experiences and task performance. The deployment of LLMs typically demands substantial computations and storage resources. For example, the LLaMA-7B (Touvron et al., 2023) model with 7 billion parameters takes up 13.5GB of memory. Moreover, the largest 65B model in the LLaMA family needs hundreds of GB for memory. Indeed, the extra-large LLMs such as GPT-3-175B (Brown et al., 2020b), OPT-175B (Zhang et al., 2022) and BLOOM-176B (Workshop et al., 2022), demand 300GB+ memory usage, making the most powerful GPUs struggling to accommodate such capacity, let alone the resource-limited edge devices. Additionally, the long input sequence lengths of LLMs further augment the computation counts with lower throughputs during inference.

Thus, it is necessary to adopt model compression techniques to reduce the resource requirement and facilitate the LLM deployment. Among them, quantization presents a promising avenue to substantially deploy LLMs on edge devices, such as mobile phones, Raspberry Pis, and FPGAs. Besides requiring fewer resources, quantization can effectively accelerate the computation with higher throughput and improve energy efficiency, by leveraging the highly efficient 8-bit fixed-point (INT8) operations on edge platforms.

Existing works primarily employ Post-Training Quantization (PTQ), which suffers from significant accuracy degradation under low-bit settings as they do not incorporate model finetuning or retraining to restore accuracy. Quantization-Aware Training (QAT) presents a promising avenue for better performance in lower-bit configurations, but its data accessibility, training cost, and acceleration issues have not been effectively addressed. For example, nearly all QAT works focus on weight-only quantization, without quantizing the floating-

point activations. Thus, they still need to utilize the floating-point operations during the computation, which can not benefit from the highly efficient lower-precision operations on edge devices (such as INT8×INT8 integer multipliers) for further speedup. The difficulty of activation quantization lies in the pronounced outliers in activations, leading to the detrimental effect and thus significant performance degradation, particularly for large model sizes. The work (Dettmers et al., 2022) demonstrates that directly setting the outliers to zero leads to a 45% performance degradation.

To deal with the above mentioned challenges, we propose EdgeQAT, the Entropy and Distribution Guided QAT, to achieve the acceleration for lightweight LLMs on the Edge. Specifically, to identify the bottleneck of activation quantization, we initiate our analysis by examining the performance degradation induced by activation quantization across various parts of the LLaMA model. We observe that the quantized query and key within the self-attention mechanism lead to the most significant accuracy loss, due to the substantial disparities between the generated attention map and its FP16 counterpart. To address this, we propose the entropy and distribution guided optimization method to mitigate the accuracy loss. In detail, we maximize the entropy of the query and key based on their distribution to equivalently minimize their quantization error. Meanwhile, we optimize the cosine similarity between the quantized and FP16 attention maps to minimize their difference with better performance. After the distribution of the quantized attention map is restored, we further introduce the token importance-aware adaptive quantization to quantize the activations with fewer bits.

By employing QAT for both weights and activations, we aim to minimize quantization error and achieve significant inference acceleration on edge devices. As training all parameters for large LLMs demands extensive GPU resources and high-quality data, we opt for lightweight LLM models in this paper for experimentation and deployment on the edge. We mainly quantize the weights and activations to 4 bits and 8 bits, following the binary nature of internal representations in computers and the trend toward advancing direct support for 4-bit operations with an increasing number of architectures (such as RISC-V). The proposed EdgeQAT can maintain state-of-the-art task performance comparable to FP16 counterparts while achieving a practical on-device speedup of up to 2.37×.

We summarize our contributions as follows:

- We design the entropy and distribution guided quantization method to mitigate information distortion in quantized query, key, and attention maps, addressing the bottleneck of QAT.
- We design the token importance-aware adaptive quantization method to quantize the activations (i.e., tokens) with fewer bits, further improving the efficiency on edge devices.
- We achieve state-of-the-art accuracy performance comparable to the FP16 model and better than other QAT methods. Our deployments across multiple edge devices demonstrate an on-device speedup of up to 2.37×.

2 Related Work

2.1 Efficient Large Language Models

Recent advancements in LLMs like GPT-4 (Achiam et al., 2023) have significantly improved NLP capabilities at the cost of massive computations and energy, limiting their accessibility and applications. This has led to the emergence of efficient and lightweight LLMs to address these limitations without compromising performance. Models such as LLaMA (Touvron et al., 2023), OPT (Zhang et al., 2022), and BLOOM (Workshop et al., 2022) offer a wide range of sizes, from as few as 125M to as many as 176B parameters, providing versatile options for various applications. To further enhance the efficiency of LLMs, multiple compression techniques have been developed (Hu et al., 2021; Frantar et al., 2022; Frantar and Alistarh, 2023; Fu et al., 2023). These methods aim to reduce model size and computational demands, enabling deployment on resource-constrained platforms such as edge devices. This shift towards more manageable models facilitates real-time NLP applications like virtual assistants and language translation, broadening the accessibility and utility of advanced NLP technologies.

2.2 Quantization for LLMs

Quantization reduces DNN bit-precision, leading to smaller models and faster inference. Current methods are divided into PTQ and QAT, each offering distinct advantages and facing unique challenges. PTQ generally results in low accuracy, especially in low-bit quantizations. To address this, Smoothquant (Xiao et al., 2023) achieves W8A8 precision by smoothing activation outliers, while ZeroQuant (Yao et al., 2022) employs a layer-by-

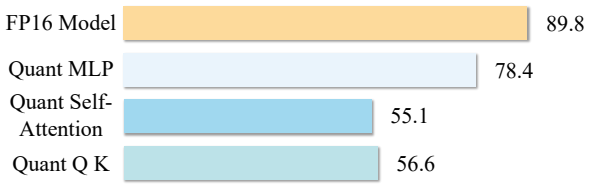

Figure 1: Accuracy analysis on the Anaphor Agr. subdataset of BLiMP with different quantized modules.

layer knowledge distillation algorithm to enhance low-bit quantization performance. Different from PTQ, QAT presents a promising avenue for better performance, requiring massive data and resources for fine-tuning, which is especially hard for LLMs.

Most PTQ and QAT works focus on weight-only quantization, and the weight-activation quantization to quantize both weights and activations is less explored. GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2023) focus on reducing the precision of weights while maintaining full-precision activations. Thus, their speedups may be limited due to the computational costs with full-precision activations. Only LLM-QAT (Liu et al., 2023) employs data-free distillation methods to quantize the weights, activations, and KV caches for large models like LLaMA-7B, which can hardly be deployed on edge devices. The exploration of weight-activation quantization with QAT for lightweight LLMs facilitating deployment on the edge is still an open field.

3 Analysis

To pinpoint the bottleneck during QAT, we analyze the performance deterioration resulting from the quantization of each part of the model. Furthermore, we explore the token importance based on the attention map and discern the importance associated with the first initial token.

3.1 Quantized Self-Attention Module

We begin by quantizing both the weights and activations for different parts of the model, including the MLP module, the whole self-attention module, or part of the self-attention (query and key), using the quantization method (Esser et al., 2019) in one-shot. When quantizing each part, other parts are not quantized. This ablation study aims to identify which components have the most detrimental impact on model performance due to quantization. The accuracy results of the LLaMA-58M model (Timiryasov and Tastet, 2023) on the Anaphor Agr. subclass of the BLiMP dataset (Warstadt et al., 2020) are

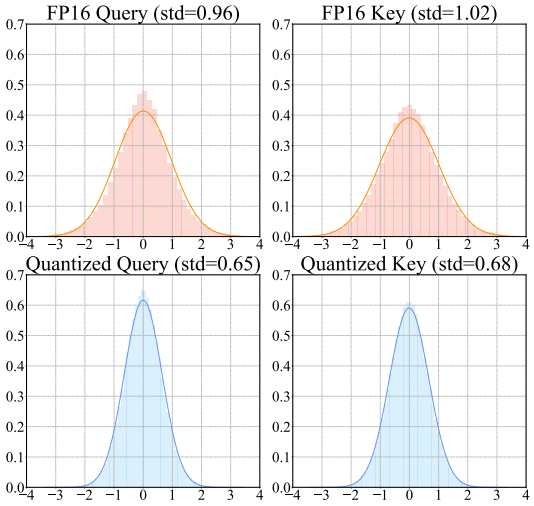

Figure 2: Distributions of query and key at the last layer of FP16 and quantized models.

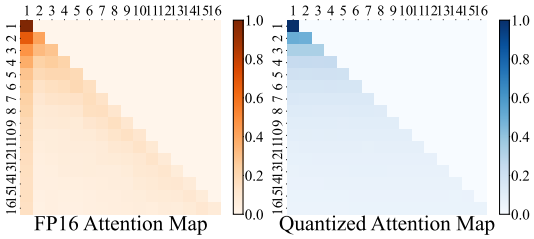

Figure 3: FP16 and quantized attention maps at the last layer of the FP16 and quantized models.

visualized in Figure 1. As observed, the quantization of the self-attention module leads to significant accuracy loss (from 89.8% to 55.1%). Among all components in self-attention, the quantization of query and key is the main reason for the substantial performance drop with 56.6% accuracy, which is close to that of quantizing the whole self-attention.

We further visualize the distributions of query and key at the last layer of quantized and FP16 models in Figure 2. As observed, the difference of the variance between the quantized and FP16 counterparts is substantial for both the query and key, inevitably leading to the deterioration of the representation capability of the attention module.

3.2 Token Importance

We visualize the heatmap of the attention map at the last layer inside the FP16 and quantized models in Figure 3. In the FP16 attention map, there is a column pattern at the first initial token, which disappears after quantization. It is evident that a significant amount of attention is allocated to the initial token. We highlight that the initial token is vital for producing text that is both coherent and

contextually meaningful. In LLMs, a distinct initial token is placed at the beginning of the input sequence, with a role in initializing the hidden layers and defining token positions within the sequence. It is visible to almost all subsequent tokens because of the nature of autoregressive language modeling. Removing certain interactions between the initial token and other tokens can fundamentally change the model generation results.

Apart from fixing the disappeared pattern with the initial token, it also raises the necessity to assess the importance of tokens with the initial token. The token importance remains valuable for further optimizations such as token pruning (Kim et al., 2024; Dong et al., 2023; Kong et al., 2022). However, as the self-attention mechanism in generative models limits each token's interaction to only those preceding it, the traditional token importance computation methods are unsuitable in LLMs.

4 Methodology

We first introduce the quantization preliminary and then explain the primary methods utilized for optimization during the training process.

4.1 Preliminary

For QAT, we adopt the symmetric quantization for both weights and activations as follows:

$$\mathcal{Q}(\mathbf{w}) = \lfloor \text{CLIP}(\frac{\mathbf{w}}{\alpha_\mathbf{w}}, -2^{b_\mathbf{w}-1}, 2^{b_\mathbf{w}-1}-1) \rceil;\ \hat{\mathbf{w}} = \mathcal{Q}(\mathbf{w}) \cdot \alpha_\mathbf{w}, \tag{1}$$

$$\mathcal{Q}(\mathbf{x}) = \lfloor \text{CLIP}(\frac{\mathbf{x}}{\alpha_\mathbf{x}}, -2^{b_\mathbf{x}-1}, 2^{b_\mathbf{x}-1}-1) \rceil;\ \hat{\mathbf{x}} = \mathcal{Q}(\mathbf{x}) \cdot \alpha_\mathbf{x}, \tag{2}$$

where $\mathbf{x}$ denotes the activations and $\mathbf{w}$ means the weights. $b_\mathbf{x}$ and $b_\mathbf{w}$ denote the bit width for the activations and weights, respectively. α denotes the trainable scale. $\lfloor \cdot \rceil$ represents rounding to the nearest integer. Thus, during the training progress, the linear projection can be calculated as follows:

$$\mathcal{F}_{Linear}(\mathbf{x}, \mathbf{w}) = \hat{\mathbf{x}} \times \hat{\mathbf{w}} = \alpha_\mathbf{x} \alpha_\mathbf{w} \left[\mathcal{Q}(\mathbf{x}) \times \mathcal{Q}(\mathbf{w}) \right]. \tag{3}$$

For the backward propagation, we use the Straight-Through Estimator (STE) (Bengio et al., 2013) to retain the derivation of the gradients:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{x}} = \frac{\partial \mathcal{J}}{\partial \hat{\mathbf{x}}} \frac{\partial \hat{\mathbf{x}}}{\partial \mathbf{x}} = \begin{cases} \frac{\partial \mathcal{J}}{\partial \hat{\mathbf{x}}}, \mathbf{x} \in \left[-2^{b_\mathbf{x}-1}, 2^{b_\mathbf{x}-1} - 1 \right] \\ 0, \quad \text{otherwise.} \end{cases} \tag{4}$$

$$\frac{\partial \mathcal{J}}{\partial \mathbf{w}} = \frac{\partial \mathcal{J}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \hat{\mathbf{w}}} \frac{\partial \hat{\mathbf{w}}}{\partial \mathbf{w}} = \begin{cases} \frac{\partial \mathcal{J}}{\partial \hat{\mathbf{x}}} \frac{\partial \mathbf{x}}{\partial \hat{\mathbf{w}}}, \mathbf{w} \in \left[-2^{b_\mathbf{w}-1}, 2^{b_\mathbf{w}-1} - 1 \right] \\ 0, \quad \text{otherwise.} \end{cases} \tag{5}$$

4.2 Entropy and Distribution Guided Optimization

Based on the analysis in Section 3.1, we conclude that the performance loss is primarily attributed to the quantized attention module with deteriorated representation capability. To address this issue, we propose the entropy and distribution guided optimization method, which statistically maximizes the entropy of representations and restores the capability of the quantized self-attention module. According to the work (Messerschmitt, 1971), for Gaussian distribution, quantizers with maximum output entropy (MOE) and minimum average error (MAE) are approximately equivalent, up to a multiplicative constant. In essence, maximizing the information entropy of quantized values is equivalent to minimizing the error caused by quantization. As observed in Figure 2, the distributions of the query $\mathbf{q}$ and the key $\mathbf{k}$ in the self-attention modules follow the Gaussian distribution as below,

$$\mathbf{q} \sim \mathcal{N}(\mu_\mathbf{q}, \sigma_\mathbf{q}), \quad \mathbf{k} \sim \mathcal{N}(\mu_\mathbf{k}, \sigma_\mathbf{k}). \tag{6}$$

The entropy can be represented as follows,

$$\mathcal{H}(\mathbf{q}) = -\sum_i p(\mathbf{q}_i) \log p(\mathbf{q}_i) = \frac{1}{2} \log 2\pi e \sigma_\mathbf{q}^2, \tag{7}$$

$$\mathcal{H}(\mathbf{k}) = -\sum_i p(\mathbf{k}_i) \log p(\mathbf{k}_i) = \frac{1}{2} \log 2\pi e \sigma_\mathbf{k}^2. \tag{8}$$

To maximize the entropy $\mathcal{H}(\mathbf{q}) \propto \sigma_\mathbf{q}^2$ and $\mathcal{H}(\mathbf{k}) \propto \sigma_\mathbf{k}^2$ during the training process, we incorporate the entropy loss $\mathcal{L}_E$ to optimize the total entropy of query and key for all layers and heads. Specifically, we re-scale the entropy loss as follows:

$$\mathcal{L}_E = -\log \left(\sum_{l=1}^{L} \sum_{h=1}^{H} \log(1 + \sigma_\mathbf{q}^2 \sigma_\mathbf{k}^2) \right), \tag{9}$$

where L and H denote the number of layers and heads, respectively. To prevent the occurrence of NaNs when scaling the loss with the log operation, we increment the deviation product by 1.

Next, we focus on fixing the distribution pattern issue in the attention map. As shown in Figure 3, the column distribution pattern with the initial tokens from the FP16 counterpart disappears after quantization in the quantized attention map. To minimize the difference between the quantized attention map and the FP16 counterpart, we introduce a distribution loss $\mathcal{L}_D$ based on the cosine similarity between the FP16 attention map $attn_f$ and quantized one $attn_q$ in each layer as follows:

$$\mathcal{L}_D = \log \left(\sum_{l=1}^{L} \sum_{h=1}^{H} \frac{attn_q \cdot attn_f}{\|attn_q\|_2 \cdot \|attn_f\|_2} \right). \tag{10}$$

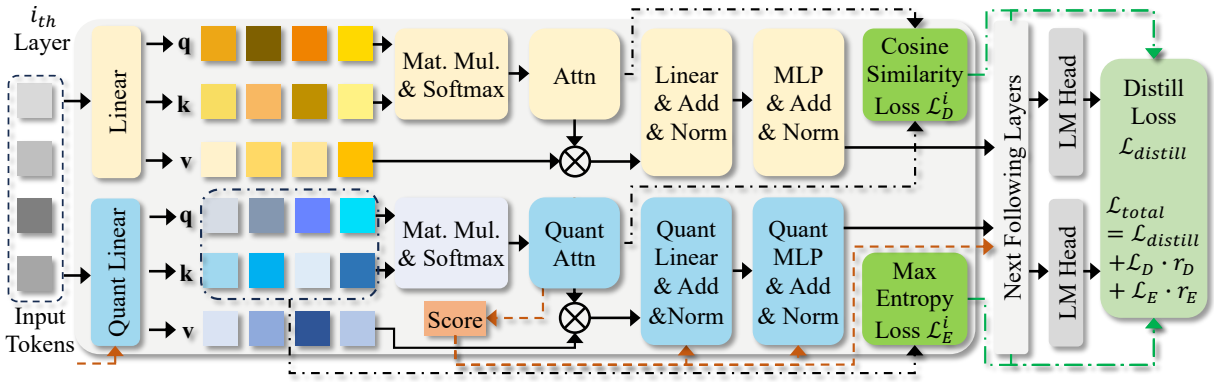

Figure 4: Overall distillation pipeline. Token adaptive QAT based on the token importance score (colored in red) with maximum entropy loss and attention map cosine similarity loss (both colored in green).

We re-scale the loss with the logarithmic operation to match the scale of the original loss.

4.3 Token Importance-Aware Adaptive Quantization

Computer specifications follow binary conventions like 8 bits, 32 bits, etc., due to the binary nature of internal representation. Recently, an increasing number of architectures are advancing direct support for 4-bit operations, as exemplified by RISC-V and similar platforms. In practice, 8-bit weight quantization is widely adopted to keep high accuracy with fast inference. However, using the same 8 bits for all quantized weights or activations is less flexible and can not make use of the innovative features with 4-bit operations in edge computing. To retain a high quantization accuracy and benefit from 4-bit operations on edge devices, we propose the token importance-aware adaptive quantization method for the quantization of activations with mixed bit widths, to dynamically assign more bits (8 bits) for important activations and fewer bits (4 bits) for unimportant ones, equivalently achieving non-power-of-two quantization with lower memory usage and faster computations compared with only using 8 bits during training and generation processes. Our algorithm is strategically aligned with the forefront of innovation in mobile computing, specifically in response to the trend set by industry leaders like Snapdragon. Tailored for seamless integration with 4-bit quantization and low-bit inference, our framework ensures compatibility and optimization for the dynamic landscape of mobile edge computing.

With the analysis in Section 3.2, we identify the token importance based on the attentivity of each token to the first initial token. Subsequently, we allocate more bits (8 bits) for quantizing important tokens, while assigning fewer bits (4 bits) for inattentive tokens. Thus, the token adaptive quantization can be represented as follows,

$$\beta(\mathbf{x}_i|\text{attn}_\mathbf{x},\rho) = \begin{cases} 8, \mathcal{F}(\mathbf{x}_i|\text{attn}_\mathbf{x},\rho) = 1 \\ 4, \mathcal{F}(\mathbf{x}_i|\text{attn}_\mathbf{x},\rho) = 0 \end{cases}, \forall i \in [1, N], \tag{11}$$

where $\mathbf{x}_i$ denotes the i^{th} token of $\mathbf{x}$ during training and generation processes. $\mathcal{F}(\mathbf{x}_i|\text{attn}_\mathbf{x},\rho)$ discriminates whether the token $\mathbf{x}_i$ is an important token based on the most recent attention map $\text{attn}_\mathbf{x}$. ρ denotes the important token ratio, i.e., ρ tokens among all tokens are set to important with $\mathcal{F}(\mathbf{x}_i|\text{attn}_\mathbf{x},\rho) = 1$ while the rest are considered unimportant. The quantization of activations can be represented as

$$\mathcal{Q}(\mathbf{x}_i) = \lfloor \text{CLIP}(\frac{\mathbf{x}_i}{\alpha_\mathbf{x}}, -2^{\beta(\mathbf{x}_i)-1}, 2^{\beta(\mathbf{x}_i)-1} - 1) \rceil, \forall i \in [1, N]. \tag{12}$$

4.4 Adaptive Training Pipeline

We visualize our training pipeline in Figure 4. We use the FP16 model (colored in yellow) to distill the quantized model (colored in blue) in QAT. We apply soft distillation, which trains a student model to mimic a teacher model by minimizing the KL divergence between their softmax outputs (Hinton et al., 2015). The distillation loss is defined as:

$$\mathcal{L}_{distill} = (1-\gamma)\mathcal{L}_{CE} + \gamma\tau^2\mathcal{L}_{KL}, \tag{13}$$

where τ is the temperature for the distillation, and γ is the coefficient balancing the KL divergence loss $\mathcal{L}_{KL}$ and the cross-entropy loss $\mathcal{L}_{CE}$. In the quantization modules, the tokens are adaptively quantized with either 8 bits or 4 bits based on their scores (colored in red) generated from the most recent attention map.

Dataset	# Bits	FP16	W8A8				W4A8				W4A4			
	Method	/	NIPQ	PACT	LSQ	Ours	NIPQ	PACT	LSQ	Ours	NIPQ	PACT	LSQ	Ours
BLiMP	Anaphor Agr.	89.8	85.5	86.4	85.4	**88.1**	58.1	86.6	85.4	**87.6**	66.2	**85.8**	82.4	85.7
	Arg. Structure	73.1	70.9	70.7	70.5	**72.2**	55.5	70.3	70.9	**72.3**	54.4	69.6	71.0	**71.3**
	Binding	72.7	71.1	71.0	70.8	**72.3**	61.7	70.6	70.9	**72.2**	51.5	68.2	71.5	**72.4**
	Control/Raising	67.5	65.5	64.6	66.1	**66.7**	54.7	64.0	65.8	**66.7**	53.6	63.6	65.4	**66.3**
	Det.-Noun Agr.	90.8	86.9	86.3	87.5	**89.2**	54.2	86.6	86.7	**89.1**	53.4	84.8	87.1	**87.5**
	Ellipsis	73.3	60.4	59.7	63.9	**69.4**	29.9	59.7	62.1	**69.8**	33.8	56.8	63.2	**65.1**
	Filler-Gap	71.8	70.2	69.0	70.2	**72.1**	66.7	69.3	69.5	**72.0**	61.1	66.8	70.2	**70.4**
	Irregular Forms	93.1	94.6	94.8	92.7	**95.0**	45.8	**95.2**	93.3	94.9	52.2	93.7	94.1	**94.9**
	Island Effects	51.2	48.2	49.2	48.2	**51.7**	43.6	50.0	48.9	**52.1**	48.5	43.3	48.2	**51.3**
	NPI Licensing	56.5	50.0	52.1	49.5	**58.3**	26.8	52.2	51.4	**57.7**	36.6	48.2	**50.9**	44.5
	Quantifiers	73.3	73.7	75.8	**82.4**	79.0	57.2	78.2	**79.4**	79.3	42.7	78.0	73.4	**80.0**
	Subj.-Verb Agr.	75.4	68.4	67.8	68.1	**73.2**	46.3	67.7	67.5	**74.0**	48.6	64.5	68.0	**71.6**
BLiMP Suppl.	Hypernym	49.3	48.0	49.0	**49.6**	48.9	49.5	48.7	48.7	**49.6**	**50.9**	50.3	49.3	50.5
	QA Congruence easy	51.6	48.4	**51.5**	46.8	50.1	35.9	50.0	46.8	**50.1**	37.5	48.4	46.8	**50.1**
	QA Congruence tricky	41.8	40.6	40.0	40.6	**41.3**	34.5	40.6	40.6	**41.3**	33.9	39.3	40.6	**41.9**
	Subj.-Aux. Inversion	88.5	**89.1**	87.9	87.3	88.5	67.8	**89.8**	83.6	89.2	54.6	87.3	85.8	**89.0**
	Turn Taking	66.1	58.2	57.1	58.9	**61.5**	43.2	57.5	59.6	**61.8**	51.4	55.7	59.2	**60.1**
	Total Average	69.7	66.5	66.6	67.0	**69.3**	48.9	66.9	66.5	**69.4**	48.9	64.9	66.3	**67.8**

Table 1: LLaMA-58M quantization results on the BLiMP dataset, including the BLiMP Supplement.

The entropy loss $\mathcal{L}_E$ and the distribution loss $\mathcal{L}_D$ (both colored in green) are added to the total loss for optimization during training as follows,

$$\mathcal{L}_{total} = \mathcal{L}_{distill} + r_E \cdot \mathcal{L}_E + r_D \cdot \mathcal{L}_D. \quad (14)$$

The ratios r_E and r_D are used to weight the entropy and distribution losses, respectively. In our experiments, we independently set $r_E = 0.5$ and $r_D = 1$ to facilitate the better optimization.

4.5 Hardware Implementations

We deploy the quantized model obtained from our method on mobile phone, Raspberry Pi, and FPGA. We integrate our computational graph into llama.cpp (Gerganov, 2023) engine on mobile and Raspberry Pi. To handle asymmetric operations, such as 4-bit & 8-bit multiplications, we utilize uniform 8-bit integer operators. The unused low-bit weights are strategically stored in byte-aligned memory units, minimizing bit wastage and addressing memory constraints on edge devices. Our hardware implementation enhances efficiency and universality for edge computing scenarios. FPGAs, with limited off-chip memory, can also benefit from our quantization schemes. We implement our inference engine with the quantization scheme based on existing LLM FPGA implementations (Chen et al., 2023). We built 4-bit and 8-bit systolic array architecture for the Multipy-Accumulate Circuit (MAC). DSP-packing is further applied to MAC for better utilization of DSP resources. For non-linear operations, we implement floating-point-based kernels.

5 Experimental Results

5.1 Experiment Setup

For the verification of our proposed methods, as the computation resources are limited, we experiment with lightweight LLMs, including LLaMA-58M (Touvron et al., 2023; Timiryasov and Tastet, 2023) and GPT2-97M (Radford et al., 2019a). We adopt the pretrain datasets from the work (Yang et al., 2023) and then perform regex-based cleaning on them. The cleaned datasets are tokenized using BytePair Encoding (BPE) with a vocabulary size of 16000. The models are then evaluated on BLiMP (Warstadt et al., 2020) for the zero-shot test and (Super)GLUE (Wang et al., 2019) for the fine-tuning test. In the absence of prior QAT studies for LLMs, we compare with well-known static quantization methods as baselines, including NIPQ (Park et al., 2022), PACT (Choi et al., 2018), and LSQ (Esser et al., 2019). The same fine-tuning recipe based on the pretrain recipe of the work (Yang et al., 2023) is adopted for our QAT method and the baselines. To make a fair comparison with the same bit width, the adaptive quantization is only adopted in Sec. 5.4 and Figure 5. Each QAT experiment is conducted on one NVIDIA TITAN RTX GPU within one day.

5.2 Hardware Deployment

We opt for the Oneplus 11 smartphone as our mobile platform, and this device is powered by the Snapdragon 8 Gen 2. All available cores have been

Method	FP16	NIPQ	PACT	LSQ	Ours
Anaphor Agr.	87.0	38.1	69.8	84.0	**84.5**
Arg. Structure	71.3	57.4	63.7	70.7	**71.7**
Binding	70.2	49.8	64.4	69.2	**69.8**
Control/Raising	66.1	54.2	62.6	65.1	**65.3**
Det.-Noun Agr.	87.4	51.4	72.3	**86.8**	86.0
Ellipsis	62.1	39.6	39.2	59.8	**59.9**
Filler-Gap	70.7	43.3	63.2	69.6	**70.4**
Irregular Forms	94.1	52.3	90.0	94.3	**95.4**
Island Effects	47.2	**59.7**	44.9	46.6	46.8
NPI Licensing	48.5	**71.3**	44.4	47.3	44.8
Quantifiers	68.0	27.5	46.7	66.1	**69.4**
Subj.-Verb Agr.	66.2	48.1	55.5	64.8	**66.0**
Average	69.9	49.4	59.7	68.7	**69.2**

Table 2: GPT2-97M Results with W4A4 Setting on the BLiMP Main Dataset.

Method	FP16	NIPQ	PACT	LSQ	Ours
CoLA	69.5	33.3	**69.3**	68.6	68.4
SST-2	87.2	49.4	**85.4**	84.6	84.1
MRPC	63.2	32.2	69.4	69.4	**69.5**
QQP	84.3	42.4	82.5	83.9	**84.1**
MNLI	72.9	35.4	67.5	70.8	**70.8**
MNLI-mm	73.7	35.8	69.1	**71.5**	71.1
QNLI	81.1	47.2	74.4	78.4	**79.4**
RTE	61.6	50.5	48.5	**57.5**	53.5
BoolQ	67.2	58.4	60.3	**63.7**	62.9
MultiRC	58.9	53.2	46.1	46.6	**54.1**
WSC	61.4	**61.4**	53.0	42.1	56.6
Average	71.0	45.4	65.9	67.0	**68.6**

Table 3: LLaMA-58M results with W4A4 setting on the (Super)GLUE dataset.

utilized for multi-thread computation. Similarly, for Raspberry Pi 5 equipped with BCM2712 quad-core Arm Cortex A76 processor, we allocate the computation among 4 cores. The latency has been reported via 100 iterations for each test based on llama.cpp. Similarly, for the FPGA evaluations, we make use of the AMD Alveo U280 FPGA from the Open Cloud Testbed (OCT), an open-source cloud platform for research (Zink et al., 2021). We implement the proposed design using the XDMA platform running at 200MHz. For the testing, we preload the inputs and parameters to the onboard HBM and measure the performance results through 10000 iterators of the accelerators.

5.3 Main Results of Model Performance

We first verify the effectiveness of our proposed EdgeQAT framework on the BLiMP (Warstadt et al., 2020) dataset with zero-shot (i.e., no fine-tuning) evaluations, and the results are shown in Table 1. We compare our method with the other three QAT works, including NIPQ, PACT, and LSQ, under different bit-width settings including W8A8 (meaning 8-bit weight and 8-bit activation quantization), W4A8, and W4A4. As observed, our proposed EdgeQAT framework achieves better performance than all other three works on the average accuracy of all subdatasets in the BLiMP dataset. Our method achieves the best performance on most of the subdatasets across three bit-width configurations. Especially for the W4A8 setting, which is the most useful in practical applications, our method achieves an average accuracy of 69.4%, which is close to the performance of the FP16 model (only 0.3% drop) and even surpasses the W8A8 setting (69.3%). For the W4A4 setting, our method maintains an average accuracy of 67.8%, showcasing a clear advantage over other methods. Only our method can achieve a competitive average accuracy close to that of the FP16 model, while the baselines usually suffer from substantial accuracy drops. NIPQ fails to restore the accuracy when the model weights are quantized to 4 bits. For PACT, it is sensitive to the bit width of the activations, as evidenced by the poor results under the W4A4 setting. The LSQ method consistently produces models with an average accuracy of about 66% to 67%, which is still lower than our method.

We also compare with the PTQ work ZeroQuant-FP (Wu et al., 2023) under the W4A8 setting. ZeroQuant-FP can achieve an average accuracy of 66.7% on BLiMP. Although it is better than the QAT works including NIPQ, PACT, and LSQ, our method still performs better than ZeroQuant-FP with non-marginal improvements.

Additionally, we deliver the evaluation results of the GPT2-97M model with the W4A4 setting to verify the generalization of our method in Table 2. We conduct the experiments on the BLiMP main dataset. Our method can achieve the highest average accuracy with the best performance on most subdatasets, demonstrating clear advantages over QAT baselines. Among the baselines struggling to restore the average accuracy, the NIPQ and PACT perform much worse with large margins.

To demonstrate the effectiveness of the EdgeQAT framework, we further finetune the models from different QAT frameworks on the (Super)GLUE dataset and show the evaluation results in Table 3. To make a fair comparison, we use the same finetuning recipe for all methods. As

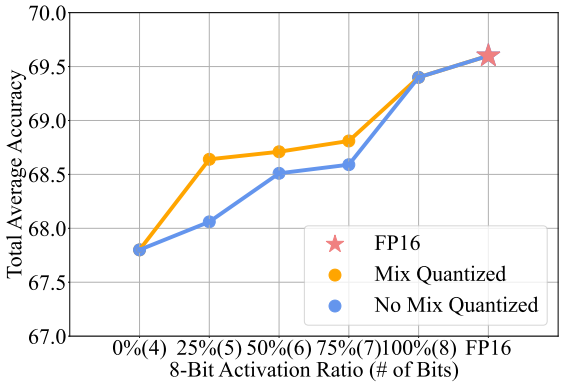

Figure 5: Adaptive quantization results with 4-bit weights and adaptive activations (4-bit or 8-bit) on the BLiMP dataset. The average accuracy is reported.

Edge	Oneplus 11		Raspberry Pi 5		AMD u280	
# Bits	ms / Token		ms / Token		ms / Token	
LLaMA-58M						
FP16	4.54	1×	15.63	1×	1.04	1×
W8A8	4.12	1.10×	9.40	1.66×	0.91	1.14×
W4A4	3.90	1.16×	6.78	2.31×	0.84	1.23×
GPT2-97M						
FP16	6.22	1×	23.04	1×	1.97	1×
W8A8	5.44	1.14×	13.75	1.68×	1.24	1.58×
W4A4	5.10	1.22×	9.74	2.37×	1.03	1.91×

Table 4: Latency results of LLaMA-58M and GPT2-97M on edge devices including the Oneplus 11 smartphone, Raspberry Pi 5, and FPGA.

observed, similarly, our method achieves the best performance in average accuracy compared with QAT baselines. While NIPQ still does not work well with more training efforts, PACT and LSQ fail to restore the average accuracy.

5.4 Adaptive Quantization Results

Following the binary conventions with binary operators (e.g., 4-bit or 8-bit multipliers) on the edge, we stick to 4-bit and 8-bit quantization settings to ensure compatibility with edge devices. We show the results of adaptive activation quantization (4 bits or 8 bits as in Equation 11) with 4 bits for weights colored with orange in Figure 5. We identify the token importance based on the first initial token of the column distribution pattern as shown in Section 3.2. Important tokens are quantized into 8 bits while 4 bits are assigned to the remaining inattentive tokens. We vary the containment ratio of 8 bits (i.e., the important token ratio ρ) from 0% to 100% to show the variance of accuracy. The accuracy of the model improves as the proportion of 8-bit activations increases, validating the effectiveness of our token importance identification method. Besides, compared with the equivalent (in terms of bits) non-power-of-two quantization (colored in blue), which uses the same bit-width for all activations (e.g., the case with 25% 8 bits and 75% 4 bits is equivalent to 5 bits for all), our adaptive method performs better, demonstrating the advantages with more bits for more important activations.

5.5 Hardware Efficiency

We show the latency results of LLaMA-58M and GPT2-97M on the Oneplus 11 smartphone, Raspberry Pi 5, and FPGA in Table 4. We can successfully achieve the acceleration of the token generation for two different models on three different devices. In particular, we can achieve 2.37× on-device speedup with W4A4 and over 1.6× speedup with W8A8 on Raspberry Pi 5, which validates the necessity and effectiveness of our proposed EdgeQAT for LLMs. Also, we achieve the acceleration on the Oneplus 11 smartphone with both W8A8 and W4A4 quantization settings, which shows the generalization of our method. As the high-end CPUs on smartphones can afford more robust floating-point processing capabilities, the acceleration attained through quantization on smartphones is not as significant as the improvements observed on the Raspberry Pi 5. For the FPGA, we achieve up to 1.9× speedup, which benefits from the flexibility of implementing custom data paths for sub-8bit operations on FPGAs.

6 Conclusion and Limitation

In this paper, we introduce EdgeQAT, an entropy and distribution guided QAT framework designed to accelerate lightweight LLMs on edge devices. We incorporate the maximum entropy theory to optimize the quantized query and key in the self-attention mechanism, and mitigate the information loss of the quantized attention map with a distribution guided loss. Besides, we adaptively quantize tokens with different bit widths based on their importance, which further reduces the average bit width for quantization. We effectively restore the model performance to that of FP16 counterparts and achieve up to 2.37× speedup on edge devices. However, we mainly experiment with lightweight LLMs due to resource constraints. We will verify our method on larger models if more data and computation resources are available.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020a. Language models are few-shot learners. *NeurIPS*, 33:1877–1901.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020b. Language models are few-shot learners.

Hongzheng Chen, Jiahao Zhang, Yixiao Du, Shaojie Xiang, Zichao Yue, Niansong Zhang, Yaohui Cai, and Zhiru Zhang. 2023. Understanding the potential of fpga-based spatial acceleration for large language model inference. *arXiv preprint arXiv:2312.15159*.

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. 2018. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Peiyan Dong, Mengshu Sun, Alec Lu, Yanyue Xie, Kenneth Liu, Zhenglun Kong, Xin Meng, Zhengang Li, Xue Lin, Zhenman Fang, et al. 2023. Heatvit: Hardware-efficient adaptive token pruning for vision transformers. In *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 442–455. IEEE.

Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. 2019. Learned step size quantization. *arXiv preprint arXiv:1902.08153*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

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

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. *arXiv preprint arXiv:2301.12726*.

Georgi Gerganov. 2023. llama.cpp: Low-latency audio streaming library for c++.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Minchul Kim, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. 2024. Token fusion: Bridging the gap between token pruning and token merging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1383–1392.

Zhenglun Kong, Peiyan Dong, Xiaolong Ma, Xin Meng, Wei Niu, Mengshu Sun, Xuan Shen, Geng Yuan, Bin Ren, Hao Tang, et al. 2022. Spvit: Enabling faster vision transformers via latency-aware soft token pruning. In *European Conference on Computer Vision*, pages 620–640. Springer.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. 2023. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*.

D. Messerschmitt. 1971. Quantizing for maximum output entropy (corresp.). *IEEE Transactions on Information Theory*, 17(5):612–612.

Sein Park, Junhyuk So, Juncheol Shin, and Eunhyeok Park. 2022. Nipq: Noise injection pseudo quantization for automated dnn optimization. *arXiv preprint arXiv:2206.00820*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019a. Language models are unsupervised multitask learners.

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

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *arXiv preprint arXiv:2308.02019*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *arXiv*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS*, 30.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Xiaoxia Wu, Zhewei Yao, and Yuxiong He. 2023. Zeroquant-fp: A leap forward in llms post-training w4a8 quantization using floating-point formats. *arXiv preprint arXiv:2307.09782*.

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

Yahan Yang, Elior Sulem, Insup Lee, and Dan Roth. 2023. Penn & BGU BabyBERTa+ for Strict-Small BabyLM Challenge. Technical report.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv*.

Michael Zink, David Irwin, Emmanuel Cecchet, Hakan Saplakoglu, Orran Krieger, Martin Herbordt, Michael Daitzman, Peter Desnoyers, Miriam Leeser, and Suranga Handagala. 2021. The open cloud testbed (oct): A platform for research into new cloud technologies. In *2021 IEEE 10th International Conference on Cloud Networking (CloudNet)*, pages 140–147. IEEE.