Benford's Law as a Distributional Prior for Post-Training Quantization of Large Language Models

Anonymous authors
Paper under double-blind review

Abstract

Post-training quantization (PTQ) is a practical way to reduce the memory footprint of large language models, but low-bit quantization is sensitive to mismatches between the quantization codebook and the empirical weight/activation distributions. We revisit Benford-like leading-digit statistics as a lightweight diagnostic of scale-broad behavior in transformer tensors. Across several model families, we observe a consistent functional dichotomy: transformational `nn.Linear` weights tend to be Benford-like, whereas `LayerNorm` and embedding parameters systematically deviate. Motivated by this observation, we propose BenQ, a data-free PTQ codebook that uses a simple log-spaced grid as a proxy for scale-broad distributions and applies it selectively to transformational layers while keeping stability-critical parameters in higher precision. In 4-bit group-wise PTQ, BenQ consistently improves over uniform RTN and is often competitive with NF4, while remaining substantially simpler than optimization-based methods. We additionally report activation quantization results as an exploratory stress test: BenQ can improve robustness over uniform baselines in some families, but performance remains mixed across models, highlighting open challenges for static-grid activation PTQ.

1 Introduction

Large Language Models (LLMs) deliver state-of-the-art results across NLP tasks, yet their memory and latency footprints hinder broad deployment (Touvron et al., 2023; Jiang et al., 2023). Post-training quantization (PTQ) is a practical remedy: by mapping full-precision weights to few-bit integers, it compresses models and often accelerates inference with modest accuracy loss (Frantar et al., 2022; Dettmers et al., 2023). The de-facto baseline, Round-To-Nearest (RTN) on a *uniform* grid, is simple and hardware-friendly, but it implicitly assumes that parameters occupy the dynamic range evenly.

Empirically, neural weights are highly non-uniform and concentrate near zero (Han et al., 2015). In low-bit regimes (e.g., 3-4 bits), uniform grids spend disproportionate capacity on rare large magnitudes while under-resolving dense near-zero regions; the mismatch is exacerbated in layers whose weight magnitudes span multiple decades. This has spurred a broad literature on *non-uniform* or *distribution-aware* quantization, from classic logarithmic level schedules (Miyashita et al., 2016) to modern per-layer, learned, or optimized codebooks for LLMs (Zhao & Yuan, 2025). In parallel, activation-aware schemes such as AWQ (Lin et al., 2024) and SmoothQuant (Lin et al., 2024) reduce sensitivity to outliers and can be combined with weight-only PTQ (Lin et al., 2024; Xiao et al., 2023).

We revisit the *Benford's Law* (BL) (Benford, 1938), a classic regularity of natural data, and show that many *transformational* layers in modern transformers (linear/attention/FFN) exhibit *Benford-like* leading-digit statistics, whereas normalization layers systematically do not, as illustrated in Figure 2. Beyond empirical evidence, we give a log-domain rationale: multiplicative stochastic optimization (SGD with decay and adaptive preconditioning) induces broad mixtures in $\log|w|$, yielding near-uniform mantissas and thus Benford-like behavior. Notably, prior work has leveraged Benford's Law as an *analysis or training signal*: Sahu et al. (2021b) propose a model as a predictor of generalization and a validation-free early-stopping

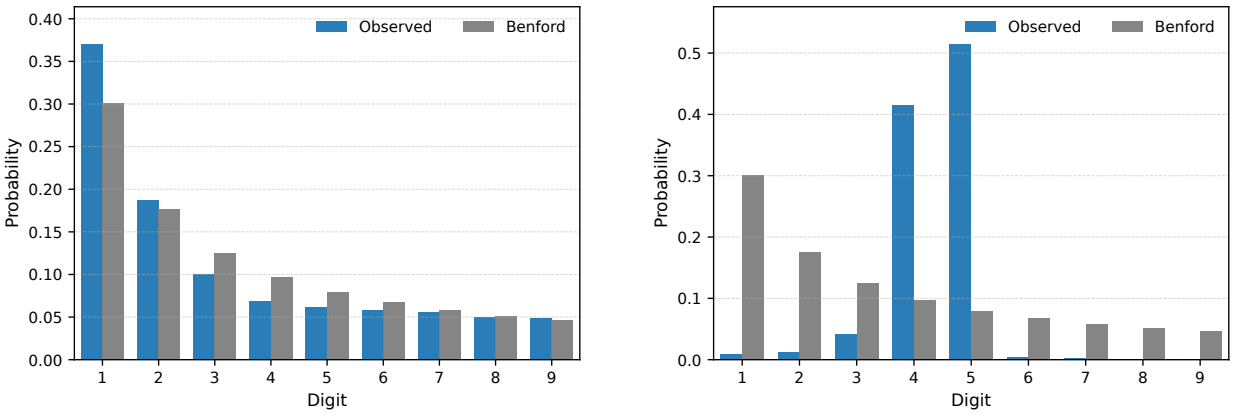
criterion, while Ott et al. (2025) regularize significant-digit histograms to improve generalization in low-data regimes.

In this work, we propose **Benford-Quant (BenQ)**, a *data-free* non-uniform PTQ codebook that replaces the linear codebook with a *log-spaced* grid *motivated* by Benford-like scale-broad statistics, and applies it *selectively* to transformational layers while leaving stability-critical parameters (e.g., LayerNorm scales, embeddings) in higher precision. We emphasize that the log grid is a simple, hardware-friendly *proxy* for scale-broad distributions, rather than a claim that it is an optimal quantizer for a specific analytic prior.

Our contributions are:

- **A diagnostic.** We measure leading-digit (Benford-like) statistics across multiple LLM families and identify a functional dichotomy: transformational `nn.Linear` weights are often Benford-like, while `LayerNorm` and embeddings deviate.
- **A simple, data-free codebook.** We introduce BenQ, a log-spaced PTQ codebook motivated by scale-broad behavior, designed as a lightweight drop-in replacement for uniform grids in group-wise PTQ.
- **Selective application.** We propose and evaluate a digit-statistics–informed selective strategy that preserves stability-critical parameters in higher precision.
- **Empirical study.** We evaluate 4-bit group-wise PTQ across several model families on perplexity and downstream benchmarks, and report activation quantization results as an exploratory stress test.

BenQ is a *data-free* codebook intended for lightweight PTQ settings. It is complementary to activation-aware or calibration-heavy methods such as AWQ (Lin et al., 2024), and SmoothQuant (Xiao et al., 2023), and to optimization-based PTQ approaches like GPTQ (Frantar et al., 2022). Accordingly, our comparisons to AWQ/GPTQ are meant to *situate* BenQ relative to stronger, calibration/optimization-based baselines.



(a) Linear Layer (`q_proj`)

(b) LayerNorm Layer

Figure 1: The dichotomy of Benford's Law compliance in Llama-3-8B. For each digit the right bar indicates Benford's expected probability, while the left bar indicates the observed one. (a) The weights of a transformational linear layer strongly adhere to the Benford distribution. (b) In contrast, the weights of a `LayerNorm` layer systematically violate the law, with their first digits overwhelmingly concentrated on a single value. This analysis motivates our selective strategy (quantize `nn.Linear`, keep `LayerNorm`/embeddings in higher precision) and supports using a log-spaced codebook as a simple proxy for scale-broad weight distributions.

2 Background and Related Work

Post-training quantization for LLMs. Post-training quantization (PTQ) compresses pretrained models by mapping full-precision weights to low-bit representations while attempting to preserve accuracy. For LLMs, practical PTQ pipelines commonly rely on group-wise scaling and simple round-to-nearest baselines (RTN), often combined with selective exclusions for stability-critical tensors (Dettmers et al., 2023).

Non-uniform codebooks. A long line of work studies non-uniform quantization levels to better match the empirical distribution of neural parameters, including logarithmic schedules (Miyashita et al., 2016; Lee et al., 2017) and more recent approaches that either optimize codebooks or adapt them to data/model statistics (Oh et al., 2021; Wu et al., 2024; Oh et al., 2022). In the LLM literature, NF4 is a prominent non-uniform 4-bit codebook motivated by a normal-quantile construction and widely used in efficient training/serving pipelines (Dettmers et al., 2023). Importantly, these methods differ in their assumptions and operational goals. Some aim for a simple static grid, while others incur optimization or calibration costs to reduce reconstruction error.

Optimization- and activation-aware PTQ. Stronger PTQ baselines explicitly optimize quantization error or incorporate activation information. GPTQ uses second-order information to minimize reconstruction loss during quantization (Frantar et al., 2022), while AWQ applies activation-aware scaling to reduce the impact of quantization on salient channels (Lin et al., 2024). Related activation-smoothing approaches such as SmoothQuant similarly aim to mitigate activation outliers (Xiao et al., 2023). Because these methods are not data-free in the same sense as static codebooks, we treat comparisons to GPTQ/AWQ primarily as context for where a lightweight codebook stands relative to calibration/optimization-based PTQ.

Benford-like statistics in neural networks. Benford's Law has been used as an analysis signal in machine learning, including as a diagnostic correlated with generalization and as a validation-free early-stopping criterion (Sahu et al., 2021b), as well as as an explicit regularizer on significant-digit histograms (Ott et al., 2025). Our work differs in goal and scope, rather than using Benford-like statistics as a training signal, we use them as a lightweight diagnostic to motivate a simple log-spaced PTQ codebook and a selective quantization strategy.

2.1 Why to Expect Benford Adherence

Benford preliminaries. Let $S_{10}(x) \in [1, 10)$ be the base-10 significand, $x = S_{10}(x) \cdot 10^k$. Benford's Law gives

$$\mathbb{P}(\lfloor S_{10}(x) \rfloor = d) = \log_{10}\big(1 + \tfrac{1}{d}\big), \quad d = 1, \ldots, 9. \tag{1}$$

A standard characterization states: x is Benford $\Leftrightarrow$ the fractional part of $\log_{10} |x|$ is uniform on $[0, 1)$ (Hill, 1995b). Thus, Benford-like behavior reduces to *equidistribution modulo* 1 in the log domain.

Multiplicative training dynamics. For a scalar weight w_t in a linear/affine map, a broad class of optimizers admits

$$w_{t+1} = (1 - \eta_t \lambda) w_t - \eta_t \phi_t g_t \;=\; M_t \, w_t + \varepsilon_t, \tag{2}$$

where $\lambda \geq 0$ (decoupled weight decay), η_t is the step size, $\phi_t > 0$ a preconditioner (e.g., Adam), g_t a stochastic gradient, and ε_t an additive residual. When $|M_t w_t|$ dominates $|\varepsilon_t|$ during nontrivial epochs, the magnitude evolves approximately multiplicatively:

$$\log |w_{t+1}| \;\approx\; \log |w_t| + \log |M_t|. \tag{3}$$

Random fluctuations in η_t, ϕ_t and data/curvature induce a noisy random walk in $\log |w_t|$, producing broad log-distributions (often close to lognormal). Aggregating across layers and training phases yields mixtures of such log-broad components.

Matrix products and Benford. Beyond temporal evolution, the *spatial structure* of neural networks also promotes Benford-like behavior. Each forward pass involves repeated matrix–vector multiplications,

$$\mathbf{h}_{\ell+1} = \mathbf{W}_\ell \mathbf{h}_\ell, \tag{4}$$

so entries of $\mathbf{h}_{\ell+1}$ are sums of products of the form $\prod_{j=1}^{k} w_j h_j$. Classical results (e.g., Hill 1995a;c) show that products of independent, non-degenerate random variables tend to produce significands that are uniformly distributed in the log domain, hence Benford. In deep networks, activations at later layers accumulate multiplicative contributions from many random weights, further broadening the log-distribution of effective coefficients. This complementary perspective explains why even static weight matrices (not only their SGD trajectories) naturally exhibit Benford-like first-digit histograms.

Consequence: near-uniform log mantissas. Broad, continuous log-distributions and random mixtures, although not an indisputable rule as stated by Berger & Hill (2019; 2011), are known mechanisms that can lead to equidistribution of $\{\log_{10}|w|\}$ modulo 1, hence Benford-like leading digits. This rationale applies to *transformational* weights (linear/attention/FFN). Parameters tightly anchored to a narrow scale—e.g., LayerNorm scales acting as learned damping factors—*violate* the broadness condition and need not be Benford, matching our empirical dichotomy.

Relation with Thermodynamics. Beyond the aforementioned motivation, Sahu et al. (2021b) justify the adherence of neural network weights to Benford's Law through an analogy between thermodynamic systems seeking equilibrium and the training process of neural networks. In this scenario, the weights are associated with particle energy states, and gradient descent is related to the system's temperature. As occurs in systems that follow Boltzmann–Gibbs statistics (e.g., an ideal gas in a closed chamber), changes in temperature induce fluctuations in the distribution of the mantissas of particle energy states around Benford's Law.

Still regarding this work, empirical analyses on the matter are also presented. In particular, using the Model Enthalpy (MLH) metric proposed by the authors, significant adherence to Benford's Law was observed across a variety of architectures, such as CNNs, LSTMs, and early pretrained Transformer-based models (e.g., BERT, ELECTRA, RoBERTa).

Implication for quantization. If $\{\log_{10}|w|\}$ is near-uniform, mass is spread across decades, with highest density near zero. A *log-spaced* grid allocates more levels where weights concentrate, reducing expected distortion at fixed bit width. Conversely, for narrow-scale parameters (e.g., LayerNorm), log spacing is suboptimal, motivating *selective* application.

Although not explicitly framed within the context of Benford's Law, there are quantization studies in the literature on the generation of logarithmic levels that demonstrate their effectiveness in better representing the distributions of weights and activations, especially in image classification tasks Oh et al. (2021); Wu et al. (2024). Moreover, logarithmic levels also stand out for their hardware efficiency, as they enable the use of optimized operations for such representations, e.g., bit-shifting Lee et al. (2017).

Note. Benford compliance (uniform mantissa in $\log_{10}$) does not imply a strictly log-uniform density; our grid is a practical proxy that captures the near-zero concentration that matters for quantization.

2.2 Baseline Quantization Methods

Post-training quantization maps a full-precision tensor $\mathbf{W} \in \mathbb{R}^{m\times n}$ to low-bit integers $\mathbf{W}_q$ via a quantizer $Q(\cdot)$ and dequantizer $DQ(\cdot)$, minimizing $\|\mathbf{W} - DQ(Q(\mathbf{W}))\|$.

Uniform RTN. A B-bit symmetric uniform quantizer uses 2^B evenly spaced levels with scale s:

$$\mathbf{W}_q = \mathrm{clip}\Big(\mathrm{round}(\mathbf{W}/s), -2^{B-1}, 2^{B-1}-1\Big), \tag{5}$$

$$DQ(\mathbf{W}_q) = \mathbf{W}_q \cdot s. \tag{6}$$

Choosing s by $\max|\mathbf{W}|/(2^{B-1}-1)$ is outlier-sensitive; group-wise scaling mitigates this by partitioning $\mathbf{W}$ and using per-group scales (Dettmers et al., 2023). This method was chosen as a way to compare the efficiency of logarithmic levels against uniform ones.

NF4 Quantization. A NF4 quantizer uses non-uniformly spaced levels with scale s as follows:

$$\mathbf{W}_q = \arg\min_{v \in \mathcal{V}_{\text{NF4}}} \left| \frac{\mathbf{W}}{s} - v \right|, \tag{7}$$

$$DQ(\mathbf{W}_q) = \mathcal{V}_{\text{NF4}}[\mathbf{W}_q] \cdot s, \tag{8}$$

where $s = max(|\mathbf{W}|)$ and $\mathcal{V}_{\text{NF4}}$ = [-1, -0.696, -0.525, -0.395, -0.284, -0.185, -0.091, 0, 0.08 , 0.161, 0.246, 0.338, 0.441, 0.563, 0.723, 1] (rounded to three decimal places) Dettmers et al. (2023). This method was selected in order to compare the logarithmic levels of BenQ with other types of non-uniform levels (namely, normal levels).

State-of-the-art Methods. We additionally compare BenQ to GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2024) to *situate* it relative to stronger calibration/optimization-based PTQ baselines. GPTQ explicitly minimizes reconstruction error using second-order information, while AWQ uses activation-aware scaling to reduce quantization error in salient channels. These methods typically achieve lower perplexity than static, data-free codebooks; our goal is not to claim superiority, but to understand where a simple log-spaced grid can be a competitive, lightweight alternative or a complementary component in hybrid pipelines.

3 The Benford-Quant Method

Benford-Quant is a post-training, data-free quantization method designed to align the quantization grid with the empirically observed logarithmic distribution of transformer weights. The method consists of three core components: (1) a distributional prior based on Benford's Law, (2) a group-wise quantization algorithm that maps weights to a non-uniform, log-spaced grid, and (3) a selective application strategy that targets only transformational layers. For transparency and reproducibility purposes, the source code will be made publicly available later.

Benford's Law as a Distributional Prior. Benford's Law (Benford, 1938) states that the probability of a number having a first significant digit $d \in \{1, \ldots, 9\}$ is given by:

$$P(d) = \log_{10}\left(1 + \frac{1}{d}\right). \tag{9}$$

This distribution arises from processes involving scale invariance and implies that values are distributed logarithmically across orders of magnitude. We use this principle as a motivating signal for scale-broad behavior in certain tensors, using it to inform the geometry of a simple log-spaced quantization grid.

Log-Uniform Quantization Grid. Motivated by Benford-like scale-broad statistics (rather than claiming an optimal codebook for Eq. (9)), we construct a non-uniform set of 2^B quantization levels, $\mathcal{L}$, that are spaced logarithmically.

For quantization with B bits, we generate $(2^{B-1}-1)$ positive levels, $\mathcal{L}^+$, within the normalized range $(\epsilon, 1]$, and include an explicit zero level. These positive levels are evenly spaced in the log domain, concentrating representational capacity near zero

$$\mathcal{L}^+ = \left\{ \exp\Big(\log(\epsilon) + i \cdot \frac{\log(1) - \log(\epsilon)}{(2^{B-1}-1)-1} \Big) \;\middle|\; i = 0, 1, \ldots, (2^{B-1}-1)-1 \right\}, \tag{10}$$

and the negative ones are described as

$$\mathcal{L}^- = \left\{ -\exp\Big(\log(\epsilon) + i \cdot \frac{\log(1) - \log(\epsilon)}{(2^{B-1}-1)} \Big) \;\middle|\; i = 0, 1, \ldots, (2^{B-1}-1) \right\}. \tag{11}$$

The full grid is then:

$$\mathcal{L} = \mathcal{L}^- \cup \{0\} \cup \mathcal{L}^+, \tag{12}$$

yielding exactly 2^B levels in $[-1, 1]$. This design inherently allocates more representational capacity to the more frequent low-magnitude weights.

The Quantization and Dequantization Procedure. The core procedure applies this non-uniform grid to a weight tensor $\mathbf{W}$ in a group-wise fashion. The full process is detailed in Algorithm 1. For each block of weights $\mathbf{w}_g$ of size G, we first compute a scale $s_g = \max(|\mathbf{w}_g|)$ to normalize the block to $[-1, 1]$. Then, each normalized weight is mapped to the index of the closest level in our static grid $\mathcal{L}$.

Algorithm 1 The Benford-Quant Quantization Procedure

Require: Weight tensor $\mathbf{W}$, bit-width B, group size G.
Ensure: Quantized indices $\mathbf{W}_q$, scales $\mathbf{S}$.

$\mathcal{L} \leftarrow \text{GenerateLogUniformLevels}(B)$ ▷ Pre-compute the 2^B non-uniform levels in $[-1, 1]$
$\mathbf{W}' \leftarrow \text{reshape}(\mathbf{W}, (-1, G))$ ▷ Reshape W into blocks of size G
Initialize empty tensors $\mathbf{W}_q$ and $\mathbf{S}$ for outputs.
for each block $\mathbf{w}_g$ in $\mathbf{W}'$ **do**
  $s_g \leftarrow \max(|\mathbf{w}_g|)$ ▷ Compute the block's scale
  $\hat{\mathbf{w}}_g \leftarrow \mathbf{w}_g / s_g$ ▷ Normalize block to $[-1, 1]$
  $\mathbf{i}_g \leftarrow \arg\min_{j \in \{1,..,2^B\}} |\hat{\mathbf{w}}_g^{\text{unsqueeze}} - \mathcal{L}_j|$ ▷ Find index of nearest level for all values in the block
  Append $\mathbf{i}_g$ to $\mathbf{W}_q$; Append s_g to $\mathbf{S}$
end for
return $\mathbf{W}_q, \mathbf{S}$

The dequantization process is a simple reversal. Given the integer indices $\mathbf{W}_q$ and the scales $\mathbf{S}$, the reconstructed weight tensor $\tilde{\mathbf{W}}$ is obtained by first performing a lookup into the level grid and then rescaling each block:

$$\tilde{\mathbf{w}}_g = \mathcal{L}[\mathbf{i}_g] \cdot s_g \tag{13}$$

where $\mathcal{L}[\mathbf{i}_g]$ denotes the element-wise lookup operation for the indices corresponding to block g.

Selective Quantization Strategy. Our empirical findings in Section 4.1 reveal that `LayerNorm` weights do not follow the logarithmic distribution assumed by our method. Applying a log-uniform quantizer to their tightly clustered, near-constant distributions is theoretically and practically suboptimal. We therefore adopt a selective quantization strategy: only `nn.Linear` layers are quantized. Critical stability layers, such as `nn.LayerNorm` and token `nn.Embedding` layers are maintained in their native FP16 precision. This surgical approach preserves model stability with a negligible memory overhead, as these layers constitute a tiny fraction of the total model parameters.

3.1 Group-Adaptive Version

In view of the original goal of the method to avoid wasting precision on values that are rarely or not used, we propose a set of quantization levels adapted to the numerical distribution of each quantized group. In this adaptation, the quantization process analyzes each weight group individually to determine its minimum value g_{min} and maximum value g_{max}, and subsequently generates a log-uniform grid centered at zero within this range. Algorithm 2 presents the described procedure in detail.

Again, the dequantization process is simply the reverse operation. That is, the tensor $\tilde{\mathbf{W}}$ is reconstructed through lookups in the grids defined by G_{min} and G_{max}.

4 Experiments and Results

Our experiments are designed to answer our core research questions. We evaluate on several transformer-based model families: Gemma3, Qwen and Qwen3, Llama3, OPT and BLOOM (Team et al., 2024; Bai

Algorithm 2 Group-Adaptive Benford-Quant

Require: Weight tensor $\mathbf{W}$, bit-width B, group size G.
Ensure: Quantized indices $\mathbf{W}_q$, scales G_{min} and G_{max}.
1: $\mathbf{W}' \leftarrow \text{reshape}(\mathbf{W}, (-1, G))$ ▷ Reshape W into blocks of size G
2: Initialize empty tensors $\mathbf{W}_q$ and G_{min}, G_{max} for outputs.
3: **for** each block $\mathbf{w}_g$ in $\mathbf{W}'$ **do**
4: $g_{min} \leftarrow \min(\mathbf{w}_g)$
5: $g_{max} \leftarrow \max(\mathbf{w}_g)$
6: $\mathcal{L} \leftarrow \text{GenerateLogUniformLevels}(B, g_{min}, g_{max})$ ▷ Log-uniform levels in $[g_{min},\ g_{max}]$
7: $\mathbf{i}_g \leftarrow \arg\min_{j \in \{1,..,2^B\}} |\hat{\mathbf{w}}_g^{\text{unsqueeze}} - \mathcal{L}_j|$ ▷ Find index of nearest level for all values in the block
8: Append $\mathbf{i}_g$ to $\mathbf{W}_q$; Append g_{min} to G_{min}; Append g_{max} to G_{max}.
9: **end for**
10: **return** $\mathbf{W}_q, G_{min}, G_{max}$

et al., 2023; Grattafiori et al., 2024; Zhang et al., 2022; Workshop et al., 2022); and on different tasks: Perplexity, LAMBADA Paperno et al. (2016), HellaSwag Zellers et al. (2019) and MMLU Hendrycks et al. (2020). All perplexity (defined as $2^{H(p)}$, where $H(p)$ is the entropy of the model's prediction) evaluations are conducted on the test split of WikiText-2 (Merity et al., 2016). The other tasks were evaluated through EleutherAI's LLM evaluation framework, namely *lm-eval*. A single computer equipped with an AMD Ryzen Threadripper 7960X 24-Cores @ 5360MHz, 256 GB of DDR5 RAM and a NVIDIA H200 GPU was used for the experiments.

4.1 RQ1: Investigating Benford's Law in Transformers

Setup. To establish the foundation for our method, we first analyze the distribution of the first significant digit for every parameter tensor in our test models. We compare the observed distribution against the theoretical Benford distribution using the Mean Absolute Deviation (MAD) metric, defined by Cerqueti & Lupi (2023) as

$$MAD = \frac{1}{9}\sum_{i=1}^{9} |p_i - b_i|, \tag{14}$$

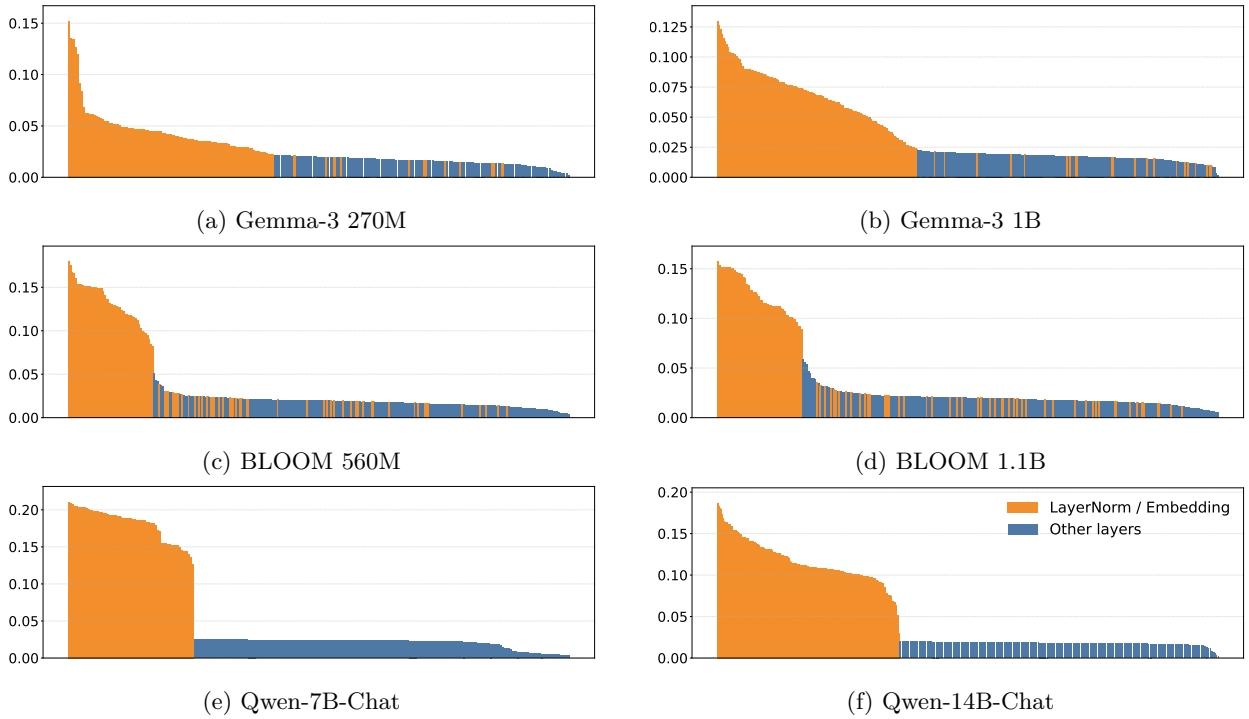
where p_i denotes the probability of digit i empirically observed, and b_i the corresponding Benford-expected probability.

Findings. Our primary finding is a strong dichotomy based on layer functionality, as illustrated in Figure 2. We consistently observe that weights from transformational `nn.Linear` layers (e.g., in attention and feed-forward blocks) closely follow Benford's distribution. This provides strong motivation for a logarithmically-spaced quantizer. In contrast, `nn.LayerNorm` weights systematically violate the law, with their values clustering around a single learned scalar (e.g., 0.35). We hypothesize these weights function not as transformations, but as learned damping factors to ensure network stability. This key finding motivates our selective quantization strategy, where `LayerNorm` layers are excluded from quantization.

Another finding is the variation in adherence to BL across model families. Figure 2 reveals a sharp split in whether layers comply with BL within the Qwen family. The same does not occur for BLOOM and Gemma3, which exhibit some outliers in their composition, i.e., normalization layers with lower MAD values than other `nn.Linear` layers. Beyond the inherent architectural differences among the models, one hypothesis for this phenomenon is related to the quality of the data used during training and/or the quality of the training process itself, given the correlation between a model's generalization capability and its compliance with BL Ott et al. (2025); Sahu et al. (2021b;a).

4.2 RQ2: Investigating BenQ's Efficiency in Transformers

Setup. Having established a potential link to logarithmic distributions in RQ1, we now validate the core design of Benford-Quant. We ask: **1.** can a Benford-inspired grid outperform the traditional uniform one?;



(a) Gemma-3 270M (b) Gemma-3 1B

(c) BLOOM 560M (d) BLOOM 1.1B

(e) Qwen-7B-Chat (f) Qwen-14B-Chat

Figure 2: Layer-wise MAD comparison across model families and sizes. Orange (hatched) represents normalization/embedding layers, while blue represents the `nn.Linear` layers (e.g., attention and feed-forward). The images reveal that weights from transformational `nn.Linear` layers closely follow Benford's distribution, while normalization/embedding weights do not.

2. is the Benford-inspired logarithmic spacing of quantization levels essential, or would any non-uniform grid suffice?; **3.** how does Benford-Quant perform in relation to state-of-the-art methods? To answer this, we conducted experiments on several transformer-based models comparing our proposed method against Uniform-RTN, NF4 Dettmers et al. (2023), GPTQ Frantar et al. (2022) and AWQ Lin et al. (2024) baselines. They are detailed in Section 2.2.

It is worth noting that RTN and NF4, like BenQ, were also implemented with selective layer quantization, i.e., keeping normalization and embedding layers unchanged. Therefore, any differences between these methods and BenQ lie in the nature of their quantization grid. In contrast, the models quantized with GPTQ and AWQ were either downloaded from HuggingFace or quantized with the AutoAWQ framework.

Findings. Given these considerations, Tables 1, 2, and 3 present the results obtained for the Llama3, Gemma3, and OPT families, respectively, under 4-bit weight quantization with a group size of 128.

Table 1: **4-bit weight-only** group-wise PTQ (group size 128) on the Llama3 family. Best quantized values are shown in bold.

	Llama-8.0B model				
	FP16	BenQ (GA)	BenQ	NF4	RTN
HellaSwag↑	0.606	0.590	0.590	**0.592**	0.590
Lambada↑	0.761	0.736	0.747	**0.751**	0.724
MMLU↑	0.655	0.626	0.629	**0.635**	0.612
Perplexity↓	6.375	7.028	7.082	**6.941**	7.372

Table 2: **4-bit weight-only** group-wise PTQ (group size 128) on the Gemma3 family. Best quantized values are shown in bold.

	Gemma3-270M model					Gemma3-1.0B model				
	FP16	BenQ (GA)	BenQ	NF4	RTN	FP16	BenQ (GA)	BenQ	NF4	RTN
HellaSwag↑	0.343	0.330	0.321	**0.331**	0.314	0.434	**0.419**	0.410	0.416	0.415
Lambada↑	0.433	0.293	0.253	**0.304**	0.152	0.435	0.308	0.313	0.328	**0.334**
MMLU↑	0.268	0.245	**0.272**	0.268	0.237	0.396	**0.367**	0.339	0.347	0.350
Perplexity↓	24.453	**36.455**	44.923	40.389	57.319	28.757	41.241	41.554	**36.047**	44.029

Table 3: **4-bit weight-only** group-wise PTQ (group size 128) on the OPT family. Best quantized values are shown in bold.

	OPT-1.3B model					OPT-2.7B model					OPT-6.7B model				
	FP16	BenQ	BenQ (GA)	NF4	RTN	FP16	BenQ	BenQ (GA)	NF4	RTN	FP16	BenQ	BenQ (GA)	NF4	RTN
HellaSwag↑	0.414	0.409	0.406	**0.410**	0.403	0.456	**0.453**	0.450	0.449	0.448	0.503	**0.497**	0.494	0.493	0.488
Lambada↑	0.588	0.573	**0.581**	0.577	0.567	0.643	0.621	0.612	**0.625**	0.617	0.677	0.671	0.665	**0.676**	0.661
MMLU↑	0.249	**0.257**	0.248	0.246	0.251	0.256	**0.262**	0.250	0.255	0.254	0.246	0.245	**0.256**	0.250	0.252
Perplexity↓	15.125	**15.753**	15.882	16.000	16.180	12.807	13.556	13.597	**13.381**	13.860	11.114	11.372	11.445	**11.294**	11.814

Discussion. Analyzing the questions raised at the beginning of this section, we observe that the Benford-inspired grid frequently outperforms the uniform baseline. For Gemma 3-270M (Table 2), for instance, the log-uniform grid yielded a 10% accuracy improvement on the LAMBADA task compared to the uniform grid. Extending this analysis to the NF4 method, a similar pattern emerges: the normal grid, evaluated on the same model, surpasses the uniform grid by 15%. The Llama3 family (Table 1) also exhibits similar behavior: for Llama3 8B, gains of approximately 2% on the LAMBADA and MMLU tasks were observed for BenQ over RTN. These results provide evidence that non-uniform levels indeed offer a more suitable representation for LLM weights.

Across most experimental settings, the log-uniform grid consistently outperformed the uniform baseline. However, although it achieved significant gains in certain non-uniform comparisons (e.g., Gemma3 1B on MMLU, where BenQ GA improved accuracy by 2% over NF4), it was not universally superior. For instance, in Llama3 8B (Table 1), NF4 quantization slightly outperformed BenQ in all tasks. Overall, these findings suggest that no single static non-uniform grid is uniformly best across architectures and tasks: NF4 can be stronger on some models (e.g., Llama3-8B), while BenQ can be favorable in others (e.g., OPT-1.3B in perplexity and MMLU). This motivates treating BenQ as a lightweight alternative/ingredient rather than a universal replacement.

Situating against GPTQ/AWQ. Table 4 situates BenQ relative to GPTQ and AWQ, which use calibration/optimization and therefore are expected to achieve lower perplexity. Indeed, GPTQ/AWQ consistently improve perplexity in these settings, while BenQ provides a simpler, data-free alternative that remains competitive on specific downstream metrics (e.g., small differences on LAMBADA for Llama3-8B). This comparison is included for context rather than as a claim that a static codebook should dominate optimized PTQ methods.

Table 4: Quantization results comparing GPTQ and AWQ against BenQ on Llama3-8B and Qwen3-4B (weight quantization). Best quantized values are shown in bold.

	Llama3-8.0B model				Qwen3-4.0B model			
	GPTQ	AWQ	BenQ (GA)	BenQ	GPTQ	AWQ	BenQ (GA)	BenQ
Lambada↑	0.731	0.745	0.736	**0.747**	**0.612**	0.588	0.591	0.593
Perplexity↓	**6.730**	6.790	7.028	7.082	**10.957**	11.105	11.771	17.217

Quantizing Activations. We extend our analysis to activation quantization, since, as discussed in Section 2.1, the spatial structure of neural networks promotes Benford-like behavior — making the log-uniform grid a favorable candidate for representing activations. Once again, we adopt 4-bit quantization with a group size of 128. Tables 5, 6, and 7 report the results for the Qwen and Qwen3, BLOOM, and Gemma3 families, respectively.

Table 5: **4-bit activation quantization** with group size 128 on Qwen/Qwen3. Best quantized values are shown in bold.

	Qwen-7B model					Qwen-14B model					Qwen3-4B model				
	FP16	BenQ	BenQ (GA)	NF4	RTN	FP16	BenQ	BenQ (GA)	NF4	RTN	FP16	BenQ	BenQ (GA)	NF4	RTN
Lambada↑	0.639	**0.601**	0.555	0.568	0.501	0.354	0.418	**0.422**	0.325	0.317	0.635	**0.557**	0.522	0.539	0.463
Perplexity↓	8.844	**9.657**	9.822	9.883	11.183	7.168	**7.575**	7.608	7.779	8.459	10.504	**12.039**	12.500	12.241	14.118

Table 6: **4-bit activation quantization** with group size 128 on BLOOM family. Best quantized values are shown in bold. † denotes divergence.

	BLOOM-560M model					BLOOM-1.0B model					BLOOM-3.0B model				
	FP16	BenQ (GA)	BenQ	NF4	RTN	FP16	BenQ (GA)	BenQ	NF4	RTN	FP16	BenQ (GA)	BenQ	NF4	RTN
HellaSwag↑	0.316	**0.306**	0.301	0.261	0.256	0.344	0.335	0.334	**0.336**	0.321	0.414	**0.396**	0.390	0.391	0.378
Lambada↑	0.353	**0.307**	0.248	0.000	0.000	0.426	**0.384**	0.372	0.290	0.196	0.517	**0.480**	0.474	0.411	0.320
MMLU↑	0.247	**0.249**	0.247	0.230	0.248	0.267	0.250	**0.263**	0.261	0.251	0.263	**0.264**	0.264	0.246	0.256
Perplexity↓	23.326	**32.910**	38.985	†	†	18.465	**21.627**	22.175	26.472	41.620	13.991	**15.642**	16.270	18.824	26.317

Table 7: **4-bit activation quantization** with group size 128 on Gemma3 family. Best quantized values are shown in bold.

	Gemma3-270M model					Gemma3-1.0B model				
	FP16	BenQ (GA)	BenQ	NF4	RTN	FP16	BenQ (GA)	BenQ	NF4	RTN
HellaSwag↑	0.343	**0.318**	0.301	0.307	0.275	0.434	**0.398**	0.372	0.380	0.329
Lambada↑	0.433	**0.306**	0.218	0.245	0.051	0.435	**0.355**	0.239	0.279	0.119
MMLU↑	0.268	**0.264**	0.248	0.257	0.246	0.396	**0.335**	0.291	0.300	0.249
Perplexity↓	24.453	**44.364**	74.880	57.352	267.430	28.757	**42.379**	57.837	49.968	118.125

Discussion (activation quantization). Across the evaluated settings, uniform RTN exhibits the largest degradation and can collapse on smaller models, confirming that naive uniform activation quantization is poorly suited to heavy-tailed activation distributions.

Non-uniform grids (NF4 and BenQ) can improve robustness over RTN, but results are mixed across model families. In BLOOM, BenQ avoids the divergence observed for NF4/RTN in some settings and yields substantially better downstream scores. In Gemma3-270M, however, all static-grid approaches remain challenging: BenQ reduces RTN collapse but still incurs large increases in perplexity, indicating that a fixed log-spaced grid is not universally sufficient for activations.

We therefore view activation quantization here as an exploratory stress test, since it highlights that scale-broad statistics alone do not fully characterize activation outliers, and that additional mechanisms (e.g., calibration, clipping, or smoothing) may be required for consistently reliable low-bit activation PTQ.

5 Limitations

This study has limitations that are important for interpreting the results.

First, BenQ uses a simple static log-spaced grid motivated by Benford-like scale-broad statistics; we do not claim that this grid is optimal for any specific analytic prior, nor do we provide a proof of optimality

(e.g., via quantile-based derivations or rate–distortion arguments). Accordingly, BenQ should be viewed as a lightweight, hardware-friendly baseline or component rather than a universal replacement for optimized PTQ methods.

Second, our evaluation focuses on a specific operating regime (4-bit, group size 128) and a fixed set of model families and benchmarks. Different bit-widths, group sizes, and deployment kernels may change the relative behavior of codebooks.

Third, comparisons to optimization- and activation-aware methods (e.g., GPTQ/AWQ) are included to situate BenQ relative to stronger calibration/optimization-based baselines. We do not claim that a static, data-free codebook should systematically outperform such methods, and we do not exhaustively tune their hyperparameters beyond standard configurations.

Finally, activation quantization remains challenging for static grids: while BenQ improves over uniform RTN in several settings, results are mixed across families and outlier behavior can still dominate. We treat the activation results as an exploratory stress test and expect that reliable low-bit activation PTQ will generally require additional mechanisms such as smoothing, clipping, or calibration.

6 Conclusion and Future Work

We conducted this study to analyze Benford's Law as a distributional prior for post-training quantization of LLMs. Initially, we presented the theoretical formulation that justifies the expectation of Benford adherence in model weights (and activations). More specifically, Benford adherence was expected for `nn.Linear` layers (MLP/Attention/FeedForward), whereas normalization and embedding layers were not expected to exhibit such behavior. Subsequently, based on the MAD metric (Equation 14), we carried out an empirical evaluation of this hypothesis, which was confirmed— as illustrated in Figure 2. We observed that the level of Benford adherence may vary across models, which can be explained by the quality of the training process, the quality of the training data, and intrinsic architectural differences.

Subsequently, we proposed BenQ, a PTQ method inspired by Benford's Law that combines selective quantization (i.e., quantizing only `nn.Linear` layers while excluding normalization and embedding layers) with a log-uniform grid. The method was evaluated against RTN, NF4, GPTQ, and AWQ. It was observed that non-uniform grids frequently outperformed the uniform baseline, suggesting that non-uniform grids can allocate quantization precision more effectively than uniform RTN in these settings. However, when comparing BenQ with NF4, it was not possible to establish a relationship of absolute superiority, as the optimal grid choice depends on the specific architecture and downstream task under consideration. Finally, when situated against state-of-the-art PTQ methods (GPTQ/AWQ), BenQ is generally weaker in perplexity—as expected for a static, data-free codebook—but can still be competitive on specific downstream metrics and serves as a simple alternative or hybrid component.

Furthermore, we report activation quantization results as an exploratory stress test. While BenQ improves over uniform RTN in several settings and can be more robust than NF4 in specific families (e.g., BLOOM), performance is mixed across models, indicating that reliable low-bit activation PTQ likely requires additional mechanisms beyond a static log-spaced grid.

Future work includes analyzing BenQ on models trained with Benford regularization Ott et al. (2025), since the method is naturally aligned with such settings and this form of regularization may further enhance the effectiveness of log-uniform quantization. Another promising direction is the hybridization of BenQ with state-of-the-art methods (e.g., AWQ, GPTQ, SmoothQuant) as well as recent multi-stage compression frameworks such as TurboQuant Zandieh et al. (2025). In this context, BenQ could act as a lightweight first-stage quantizer for tensors exhibiting Benford-like, scale-broad statistics, while a second residual- or inner-product-preserving stage could compensate for the limitations of a static log-spaced grid in more sensitive scenarios, particularly for activations and cache-like representations. Finally, extending the analysis of BenQ to architectures of different natures—such as Vision Transformers and TSFMs—remains an important direction for future work.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Frank Benford. The law of anomalous numbers. *Proceedings of the American philosophical society*, pp. 551–572, 1938.

Arno Berger and Theodore Hill. Benford's law strikes back: No simple explanation in sight for mathematical gem. *The Mathematical Intelligencer*, 33:85–91, 03 2011. doi: 10.1007/s00283-010-9182-3.

Arno Berger and Theodore P. Hill. The mathematics of benford's law: a primer. *Statistical Methods & Applications*, pp. 1–17, 2019. URL https://api.semanticscholar.org/CorpusID:202583554.

Roy Cerqueti and Claudio Lupi. Severe testing of benford's law. *Test*, 32(2):677–694, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023. *URL https://arxiv. org/abs/2305.14314*, 2, 2023.

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.

Aaron Grattafiori, , et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *CoRR*, abs/2009.03300, 2020. URL https://arxiv.org/abs/2009.03300.

Theodore P Hill. Base-invariance implies benford's law. *Proceedings of the American Mathematical Society*, 123(3):887–895, 1995a.

Theodore P Hill. The significant-digit phenomenon. *The American Mathematical Monthly*, 102(4):322–327, 1995b.

Theodore P Hill. A statistical derivation of the significant-digit law. *Statistical science*, pp. 354–363, 1995c.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Edward H Lee, Daisuke Miyashita, Elaina Chai, Boris Murmann, and S Simon Wong. Lognet: Energy-efficient neural networks using logarithmic computation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5900–5904. IEEE, 2017.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016. URL https://arxiv.org/abs/1609.07843.

Daisuke Miyashita, Edward H Lee, and Boris Murmann. Convolutional neural networks using logarithmic data representation. *arXiv preprint arXiv:1603.01025*, 2016.

Sangyun Oh, Hyeonuk Sim, Sugil Lee, and Jongeun Lee. Automated log-scale quantization for low-cost deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 742–751, 2021.

Sangyun Oh, Hyeonuk Sim, Jounghyun Kim, and Jongeun Lee. Non-uniform step size quantization for accurate post-training quantization. In *European Conference on Computer Vision*, pp. 658–673. Springer, 2022.

Julius Ott, Huawei Sun, Enrico Rinaldi, Gianfranco Mauro, Lorenzo Servadei, and Robert Wille. Exploiting benford's law for weight regularization of deep neural networks. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=TnT59yz7lc.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 1525–1534, 2016.

Surya Kant Sahu, Abhinav Java, and Arshad Shaikh. On the connection of benford's law and neural networks. *CoRR*, 2021a.

Surya Kant Sahu, Abhinav Java, Arshad Shaikh, and Yannic Kilcher. Rethinking neural networks with benford's law, 2021b. URL https://arxiv.org/abs/2102.03313.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

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

Zhuguanyu Wu, Jiaxin Chen, Hanwen Zhong, Di Huang, and Yunhong Wang. Adalog: Post-training quantization for vision transformers with adaptive logarithm quantizer. In *European Conference on Computer Vision*, pp. 411–427. Springer, 2024.

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

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. Turboquant: Online vector quantization with near-optimal distortion rate. *arXiv preprint arXiv:2504.19874*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Pengxiang Zhao and Xiaoming Yuan. Ganq: Gpu-adaptive non-uniform quantization for large language models. *arXiv preprint arXiv:2501.12956*, 2025.