OAC: Output-adaptive Calibration for Accurate Post-training Quantization

Ali Edalati[1], Alireza Ghaffari[1,2], Mahsa Ghazvini Nejad[1], Lu Hou[1], Boxing Chen[1], Masoud Asgharian[2], Vahid Partovi Nia[1]

[1]Huawei Noah's Ark Lab
[2]Department of Mathematics and Statistics, McGill University
vahid.partovinia@huawei.com

Abstract

Deployment of Large Language Models (LLMs) has major computational costs, due to their rapidly expanding size. Compression of LLMs reduces the memory footprint, latency, and energy required for their inference. Post-training Quantization (PTQ) techniques have been developed to compress LLMs while avoiding expensive re-training. Most PTQ approaches formulate the quantization error based on a layer-wise Euclidean loss, ignoring the model output. Then, each layer is calibrated using its layer-wise Hessian to update the weights towards minimizing the quantization error. The Hessian is also used for detecting the most salient weights to quantization. Such PTQ approaches are prone to accuracy drop in low-precision quantization. We propose Output-adaptive Calibration (OAC) to incorporate the model output in the calibration process. We formulate the quantization error based on the distortion of the output cross-entropy loss. OAC approximates the output-adaptive Hessian for each layer under reasonable assumptions to reduce the computational complexity. The output-adaptive Hessians are used to update the weight matrices and detect the salient weights towards maintaining the model output. Our proposed method outperforms the state-of-the-art baselines such as SpQR and BiLLM, especially, at extreme low-precision (2-bit and binary) quantization.

Extended version — https://arxiv.org/pdf/2405.15025

1 Introduction

The development of Large Language Models (LLMs) has sparked a revolution in natural language processing, leading to remarkable advancements in various tasks, including but not limited to reasoning, question answering, text generation, and few-shot learning (Radford et al. 2019; Brown et al. 2020; Zhang et al. 2022; Touvron et al. 2023a,b; Achiam et al. 2023). LLMs comprise a huge number of parameters, exceeding tens and even hundreds of billions, which is considered one of the key factors to their success (Brown et al. 2020). Their large size, however, is associated with major computational complexities and deployment barriers, especially, on resource-limited machines.

Post-training Quantization (PTQ) approaches have been introduced to overcome LLMs deployment challenges by reducing the precision of the model weights or activations while maintaining the performance without extra training (Xiao et al. 2023; Yao et al. 2022; Frantar et al. 2023; Lin et al. 2024; Kim et al. 2024; Dettmers et al. 2024; Huang et al. 2024). PTQ techniques eliminate the necessity of Quantization-aware Training (QAT) methods (Kim et al. 2023; Liu et al. 2024; Shao et al. 2024; Du et al. 2024) that are computationally demanding for LLMs. PTQ methods are one of the main techniques for efficient deployment of LLMs, enabling near loss-less compression up to 4 bits (Dettmers et al. 2024). However, extremely low-precision (2-bit or binary) PTQ is still an open challenge, which is the main focus of our proposed method.

To recover the performance of the quantized model, most novel PTQ methods (Frantar et al. 2023; Chee et al. 2023; Dettmers et al. 2024; Huang et al. 2024) perform a layer-wise calibration to reduce the quantization error. These methods measure the quantization error by the ℓ_2 loss between the output of the original and quantized layers over a small calibration set as shown in Figure 1:a). We refer to such approaches as *output-agnostic calibration* because they only consider the output of the individual linear layers and ignore the model output. The output-agnostic methods compute the second derivative of the ℓ_2 loss for the weights of each layer to formulate a layer-wise Hessian. The Hessian is used to measure the saliency of weights in order to detect the most salient ones as outliers (Dettmers et al. 2024; Kim et al. 2024; Huang et al. 2024). Also, according to the Hessian, the weights are updated toward minimizing the ℓ_2 error (Frantar et al. 2023; Chee et al. 2023; Dettmers et al. 2024; Huang et al. 2024). Given their output-agnostic nature, they are deemed inadequate to recover the performance of the original model at extreme low-precision quantization.

We introduce **Output-adaptive Calibration (OAC)** to directly minimize the distortion of the model output after quantization, as shown in Figure 1:b). OAC reduces the quantization error of the model output and significantly outperforms the state-of-the-art PTQ methods at extreme low-precision quantization such as 2-bit and binary PTQ.

OAC follows the layer-by-layer recipe to quantize and calibrate the linear layers of LLMs. Our proposed method, however, employs the second derivative of the output cross-entropy loss to compute the *output-adaptive Hessian* in contrast to the existing PTQ methods that use the Hessian of

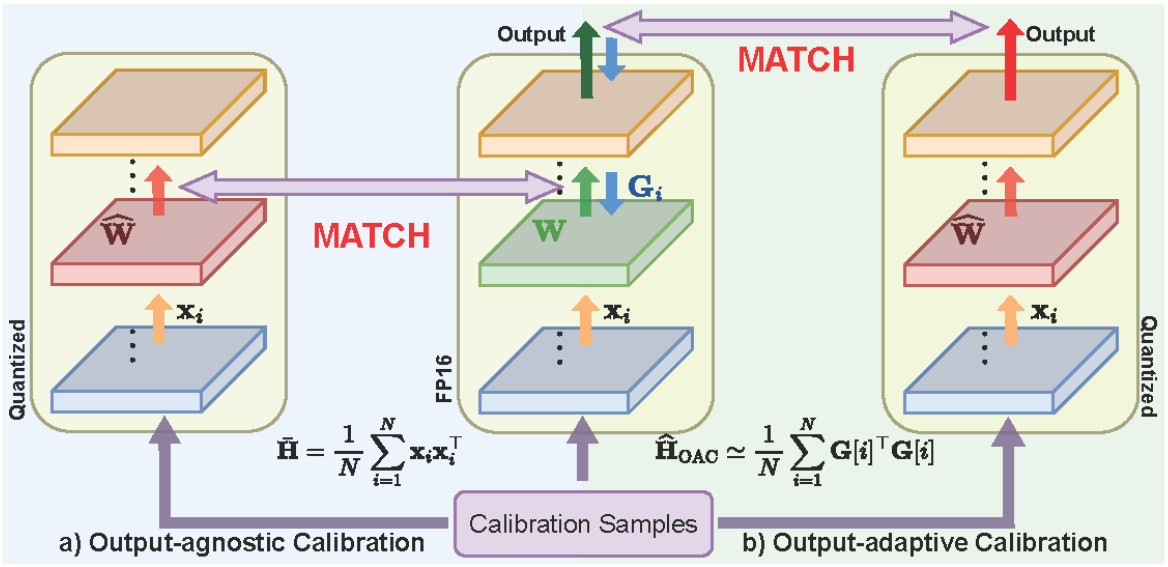

Figure 1: a) Output-agnostic calibration minimizes the ℓ_2 loss between the output of the quantized and original layers. b) Output-adaptive calibration matches the final output of the original and quantized models.

the layer-wise ℓ_2 loss. We utilize the output-adaptive Hessian for updating the weights and measuring their saliency to accurately calibrate the model. The computation of the exact output-adaptive Hessian for large models is, however, computationally infeasible. To this end, we employ several techniques to reduce the computational complexity of approximating the output-adaptive Hessian.

Our proposed method, OAC, adopts the common assumptions in the PTQ literature (Frantar and Alistarh 2022; Frantar et al. 2023; Dettmers et al. 2024), e.g. (i) the independence of linear layers, and (ii) the independence of the rows of the weight matrices. Assumption (i) is used for computation of the output-adaptive Hessian matrix of each layer, while assumption (ii) is used, in conjunction with the Fisher Information Identity, towards approximating the row-wise Hessians. Moreover, we formulate our quadratic optimization problem by aggregating the row-wise Hessian matrices to significantly reduce the memory footprint.

We apply our proposed method, OAC, for extreme low-precision PTQ of several LLMs and evaluate their performance on various tasks. OAC outperforms all of the state-of-the-art PTQ methods in 2-bit and binary PTQ of LLMs, by a significant margin.

To summarize, our main contributions are:

- We propose OAC, a novel output-adaptive calibration method for PTQ of LLMs that outperforms the state-of-the-art, especially in extreme low-precision quantization. OAC achieves superior results for binary and 2-bit quantization. To the best of our knowledge, this is the first output-adaptive calibration method for LLMs.
- We propose a quantization technique that minimizes the layer-wise quantization effect on the output cross-entropy loss. To achieve this, we develop an efficient Hessian approximation technique as the core component of our method. Integrating our proposed output-adaptive Hessian into other Hessian-based PTQ methods improves the accuracy.
- We provide theoretical insights on how to approximate the Hessian and provide a thorough experimental evaluation of various tasks to support our proposed method.

2 Related Work

There are two main categories of quantization techniques notably known as Quantization-aware Training (QAT) and Post-training Quantization (PTQ). QAT methods perform the quantization and training simultaneously (Liu et al. 2024; Jacob et al. 2018; Li et al. 2017; Shao et al. 2024). Considering the computational costs of re-training large models, QAT techniques are quite costly. The viable alternative is employing PTQ techniques, which utilize accurate solvers to minimize the quantization error without further training. Many PTQ methods leverage calibration that slightly modifies the model weights to reduce the quantization error on a small calibration set (Gholami et al. 2022).

Among the existing PTQ methods, AdaRound (Nagel et al. 2020), OBQ (Frantar and Alistarh 2022), AdaQuant (Hubara et al. 2021), and BRECQ (Li et al. 2021) are designed and applied to small computer vision models with around 100 million parameters. However, applying these methods to LLMs with billions of parameters is computationally intensive.

ZeroQuant (Yao et al. 2022), LLM.int8() (Dettmers et al. 2022), and SmoothQuant (Xiao et al. 2023) are among the first techniques that investigate PTQ of LLMs. ZeroQuant explores the quantization granularity in addition to layer-wise knowledge distillation. LLM.int8() separates the outlier activations using a threshold while quantizing the model to INT8 format. SmoothQuant (Xiao et al. 2023) reduces

the activation quantization complexity by scaling the inputs of layers. Despite succeeding at 8-bit quantization of LLMs, ZeroQuant, SmoothQuant, and LLM.int8() perform poorly on extreme low-precision quantization such as 2-bit.

Inspired by (Hassibi and Stork 1992; LeCun, Denker, and Solla 1989), OPTQ (Frantar et al. 2023) performs a column-wise calibration procedure on the weights, which enables more accurate 3- and 4-bit PTQ of LLMs using a reasonable budget on time and computational resources. AWQ (Lin et al. 2024) detects the salient weights and scales them to reduce the PTQ error. QuIP (Chee et al. 2023) uses a generalized version of the OPTQ update formula in addition to incoherence pre-processing of the weights and Hessians to reduce the accuracy drop at sub-4-bit PTQ. QuIP# (Tseng et al. 2024) enhances QuIP by randomized Hadamard transform (Halko, Martinsson, and Tropp 2011), vector quantization, and fine-tuning. SpQR (Dettmers et al. 2024) improves the OPTQ calibration strategy with two major modifications to achieve near loss-less compression of LLMs up to 4-bits, (i) detecting and isolating the outliers by the Hessian (ii) performing a second round of quantization on the quantization parameters such as scales and zeros to reduce the average bit-width while keeping the outliers at FP32 format and using small group quantization.

SqueezeLLM (Kim et al. 2024) investigates the non-uniform PTQ of LLMs using sensitivity-based K-means clustering without any calibration. To detect the salient weights, SqueezeLLM approximates the Hessian of each linear layer considering the output loss using the Fisher Information Identity. However, the non-uniform quantization is often associated with deployment challenges during the inference (Gholami et al. 2022). OmniQuant (Shao et al. 2024) introduced an efficient QAT method that freezes the model weights while learning the quantization parameters to achieve the accuracy of QAT methods with a light training recipe. Orthogonal to our work, AQLM (Egiazarian et al. 2024) uses Additive Quantization (Babenko and Lempitsky 2014) as well as fine-tuning to overcome low-precision quantization. PB-LLM (Yuan, Shang, and Dong 2024) investigated partial binarization of LLMs through a two-stage quantization recipe, (i) performing an OPTQ-based PTQ method on the non-salient binarized weights, (ii) performing QAT on the quantized model to further recover the accuracy while the salient weights are frozen. BiLLM (Huang et al. 2024) developed a more accurate PTQ method for the binarization of LLMs by identifying and structurally selecting salient weights. Furthermore, BiLLM minimizes the quantization error using a binary residual approximation strategy. BiLLM also employs splitting search to group and quantizes non-outlier weights based on their bell-shaped distribution.

Our proposed method has the following advantages compared to the state-of-the-art PTQ methods.

- Our method computes the output-adaptive Hessian to calibrate the weights considering the model output cross-entropy loss, while OPTQ, QuIP, QuIP#, SpQR, and BiLLM use the response-agnostic layer-wise Hessian.
- In contrast to SqueezeLLM and BRECQ, our method does not rely on the assumption that the output-adaptive Hessian is diagonal, and achieves a more accurate approximation. Moreover, BRECQ cannot be scaled to LLMs due to its computational complexity. Non-uniform quantization of SqueezeLLM is associated with deployment challenges and also fails to support extreme low-precision quantization.

3 Output-agnostic Calibration Background

In this section, we describe our notation as well as the response-agnostic calibration setting which is used by the status quo PTQ methods (Frantar and Alistarh 2022; Frantar et al. 2023; Chee et al. 2023; Dettmers et al. 2024).

Consider an LLM, trained using the cross-entropy (CE) loss. Let $\boldsymbol{\theta} \in \mathbb{R}^D$ be a D-dimensional vector of model weights comprising the weight matrices $\mathbf{W}^{(l)} \in \mathbb{R}^{d_{\text{row}} \times d_{\text{col}}}, l = 1, \ldots, L$ i.e. $\boldsymbol{\theta} = \text{vec}[\mathbf{W}^{(l)}]_{l=1}^{L}$, where L is the total number of layers. Let $\mathbf{y} \in \mathbb{R}^{d_{\text{vocab}}}$ denote the model output given the input $\mathbf{x} \in \mathbb{R}^{d_{\text{col}}}$. To simplify the notation, we drop l from $\mathbf{W}$ when there is no risk of confusion. Also, $\mathbf{x}^{(l)} \in \mathbb{R}^{d_{\text{col}}}$ denotes the input of the l^{th} layer. In what follows, $\mathbf{A}_{:,q}$, $\mathbf{A}_{q,:}$, and $\mathbf{A}_{j,k}$ denote the q^{th} column, q^{th} row, and the $(j,k)^{\text{th}}$ element of the matrix $\mathbf{A}$, respectively. Most PTQ methods, follow a layer-by-layer recipe to quantize and calibrate the linear layers of LLMs.

Let $\mathbf{W}$ denote the weight matrix of the l^{th} linear layer. The quantization and calibration of $\mathbf{W}$ is modeled by adding a small update term, $\delta\mathbf{W}$, such that $\widehat{\mathbf{W}} = \mathbf{W} + \delta\mathbf{W}$. To accurately quantize the weights *after training*, it is necessary to define a loss function to measure the quantization error. In the response-agnostic setting (Frantar and Alistarh 2022; Frantar et al. 2023; Chee et al. 2023; Dettmers et al. 2024), the quantization error, ε_{ℓ_2}, is measured by the ℓ_2 loss between the output of the quantized and original layer

$$\begin{aligned}\varepsilon_{\ell_2} &= \mathbb{E}_{\mathbf{x}^{(l)}}\left[\|\mathbf{W}\mathbf{x}^{(l)} - \widehat{\mathbf{W}}\mathbf{x}^{(l)}\|_2^2\right] \\ &= \mathbb{E}_{\mathbf{x}^{(l)}}\left[\text{tr}(\delta\mathbf{W}\mathbf{x}^{(l)}\mathbf{x}^{(l)^\top}\delta\mathbf{W}^\top)\right] \\ &= \text{tr}(\delta\mathbf{W}\bar{\mathbf{H}}\delta\mathbf{W}^\top), \end{aligned} \tag{1}$$

where $\bar{\mathbf{H}} = \mathbb{E}_{\mathbf{x}^{(l)}}[\mathbf{x}^{(l)}\mathbf{x}^{(l)^\top}]$ is the response-agnostic Hessian and it is assumed that the rows of the weight matrix independently contribute to ε_{ℓ_2} (Frantar and Alistarh 2022).

Next, an iterative calibration procedure is performed on the columns of $\mathbf{W}$ to minimize the quantization error (Frantar and Alistarh 2022; Frantar et al. 2023; Dettmers et al. 2024). At the q^{th} iteration, $\mathbf{W}_{:,q}$ is quantized to $\widehat{\mathbf{W}}_{:,q}$, and the remaining columns are updated to minimize ε_{ℓ_2}. Thus, the problem is formulated as the following optimization

$$\begin{aligned} \underset{\delta\mathbf{W}}{\arg\min} \quad & \text{tr}(\delta\mathbf{W}\bar{\mathbf{H}}\delta\mathbf{W}^\top), \\ \text{s.t.} \quad & \delta\mathbf{W}_{:,q} = \widehat{\mathbf{W}}_{:,q} - \mathbf{W}_{:,q}. \end{aligned} \tag{2}$$

Having solved the optimization problem (2) according to (Frantar and Alistarh 2022; Frantar et al. 2023), the optimal update at the q^{th} iteration is

$$\delta\mathbf{W}_q^* = -\frac{\mathbf{W}_{:,q} - \widehat{\mathbf{W}}_{:,q}}{[\bar{\mathbf{H}}^{-1}]_{q,q}}[\bar{\mathbf{H}}^{-1}]_{q,:}. \tag{3}$$

The most salient weights are labeled as outliers. However, various outlier mitigation techniques have been developed (Dettmers et al. 2024; Kim et al. 2024; Huang et al. 2024; Gholami et al. 2022). Equation (4) is used to measure the saliency of each weight

$$s_{j,k} = \frac{(\mathbf{W}_{j,k} - \widehat{\mathbf{W}}_{j,k})^2}{[\bar{\mathbf{H}}^{-1}]_{k,k}}. \tag{4}$$

4 Proposed Output-adaptive Calibration

In contrast to the existing output-agnostic approach, we propose calibrating the weights toward minimizing the difference between the model output loss before and after quantization, hence the reason for the name **Output-adaptive Calibration (OAC)**. In other words, we measure the quantization error based on the distortion of the cross-entropy loss i.e. $\mathcal{L}_{\text{CE}}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$ after quantizing the weights, where $\boldsymbol{\theta} = \text{vec}[\mathbf{W}^{(l)}]_{l=1}^{L}$. The quantization is denoted by $\widehat{\boldsymbol{\theta}} = \boldsymbol{\theta} + \delta\boldsymbol{\theta}$ in which $\delta\boldsymbol{\theta}$ is the update term. Therefore, the OAC error is

$$\begin{aligned} \varepsilon_{\text{OAC}} &= \mathbb{E}_{(\mathbf{x},\mathbf{y})}[\mathcal{L}_{\text{CE}}(\mathbf{x}, \mathbf{y}, \widehat{\boldsymbol{\theta}}) - \mathcal{L}_{\text{CE}}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})] \\ &= \delta\boldsymbol{\theta}^\top \bar{\mathbf{g}} + \delta\boldsymbol{\theta}^\top \bar{\mathbf{H}}_{\text{OAC}}^{(\boldsymbol{\theta})} \delta\boldsymbol{\theta} + \mathcal{O}(\|\delta\boldsymbol{\theta}\|^3), \end{aligned} \tag{5}$$

where $\bar{\mathbf{H}}_{\text{OAC}}^{(\boldsymbol{\theta})} = \mathbb{E}_{(\mathbf{x},\mathbf{y})}\left[\frac{\partial^2 \mathcal{L}_{\text{CE}}}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^\top}\right] \in \mathbb{R}^{D\times D}$ is the output-adaptive Hessian of the entire weights and $\bar{\mathbf{g}} = \mathbb{E}_{(\mathbf{x},\mathbf{y})}\left[\frac{\partial \mathcal{L}_{\text{CE}}}{\partial\boldsymbol{\theta}}\right] \simeq 0$ since the model is trained (Nagel et al. 2020). For LLMs, the computation of $\bar{\mathbf{H}}_{\text{OAC}}^{(\boldsymbol{\theta})}$ is infeasible due to its huge size $\mathcal{O}(D^2)$. In the next sections, we illustrate how our proposed approach circumvents the computation of exact $\bar{\mathbf{H}}_{\text{OAC}}^{(\boldsymbol{\theta})}$.

Cross-layer Independence

Following the layer-wise quantization and calibration recipe of our proposed method, we assume that the linear layers are independent and their output-adaptive Hessians can be computed independently. Therefore, $\bar{\mathbf{H}}_{\text{OAC}}^{(\boldsymbol{\theta})}$ is approximated by a block-diagonal matrix, where the l^{th} non-zero diagonal block, denoted by $\bar{\mathbf{H}}_{\text{OAC}}^{(l)}$, corresponds to the output-adaptive Hessian of the l^{th} linear layer, as shown in Figure 2:1). To further simplify the notation, we drop l from $\bar{\mathbf{H}}_{\text{OAC}}^{(l)}$ when there is no risk of confusion. Having assumed the cross-layer independence, the quantization error during the calibration of each linear layer is computed as

$$\begin{aligned} \varepsilon_{\text{OAC}} &= \mathbb{E}_{(\mathbf{x},\mathbf{y})}[\mathcal{L}_{\text{CE}}(\mathbf{x}, \mathbf{y}, \widehat{\boldsymbol{\theta}}) - \mathcal{L}_{\text{CE}}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})] \\ &\simeq \delta\mathbf{w}^\top \bar{\mathbf{H}}_{\text{OAC}} \delta\mathbf{w}, \end{aligned} \tag{6}$$

where $\mathbf{w} = \text{vec}(\mathbf{W})$, $\bar{\mathbf{H}}_{\text{OAC}} = \mathbb{E}_{(\mathbf{x},\mathbf{y})}\left[\frac{\partial^2 \mathcal{L}_{\text{CE}}}{\partial\mathbf{w}\partial\mathbf{w}^\top}\right] \in \mathbb{R}^{d_{\text{row}}d_{\text{col}} \times d_{\text{row}}d_{\text{col}}}$ is the output-adaptive Hessian of the linear layer, and $\delta\mathbf{w} = \text{vec}(\widehat{\mathbf{W}} - \mathbf{W})$ is the update term.

Considering the dimensions of linear layers in LLMs, $\bar{\mathbf{H}}_{\text{OAC}}$ is a huge matrix $\mathcal{O}(d_{\text{row}}^2 d_{\text{col}}^2)$ with memory-intensive computation. Some existing methods, such as (Kim et al. 2024; Li et al. 2021), ignore all of the inter-element correlations and assume that all of the weights are independent. Such methods approximate the Hessian of each layer by a diagonal matrix comprising the diagonal elements of the Fisher information matrix. However, this over-restrictive assumption leads to accuracy drop (Hassibi and Stork 1992). To achieve a more accurate Hessian approximation, we relax this assumption as described in the next section.

Cross-row Independence

In a linear layer, the output elements are independently generated by the inner product of the rows of the weight matrix and the input columns. In our proposed method, OAC, we assume that the rows of $\mathbf{W}$ are independent. Consequently, each row of $\mathbf{W}$ has its own row-wise output-adaptive Hessian that is computed separately. Therefore, $\bar{\mathbf{H}}_{\text{OAC}}$ is approximated by a block diagonal matrix, where the j^{th} block, $\bar{\mathbf{H}}_{\text{OAC}_j}$, corresponds to the Hessian of $\mathbf{W}_{j,:}$ as shown in Figure 2:2).

The rows of $\mathbf{W}$ are assumed to be independent and the columns of $\mathbf{W}$ are iteratively calibrated (Hassibi and Stork 1992; Frantar and Alistarh 2022; Frantar et al. 2023), therefore, the optimal update is found by solving

$$\begin{aligned} \underset{\delta\mathbf{W}}{\arg\min} \quad & \delta\mathbf{W}^\top \bar{\mathbf{H}}_{\text{OAC}} \delta\mathbf{W} \simeq \sum_{j=1}^{d_{\text{row}}} \delta\mathbf{W}_{j,:} \bar{\mathbf{H}}_{\text{OAC}_j} \delta\mathbf{W}_{j,:}^\top \\ \text{s.t.} \quad & \delta\mathbf{W}_{:,q} = \widehat{\mathbf{W}}_{:,q} - \mathbf{W}_{:,q}, \end{aligned} \tag{7}$$

where only the row-wise Hessians, $\bar{\mathbf{H}}_{\text{OAC}_j}$, should be computed. Therefore, the memory footprint is reduced from $\mathcal{O}(d_{\text{row}}^2 d_{\text{col}}^2)$ to $\mathcal{O}(d_{\text{row}} d_{\text{col}}^2)$. However, $\mathcal{O}(d_{\text{row}} d_{\text{col}}^2)$ is still quite huge compared to the output-agnostic ℓ_2 Hessian $\mathcal{O}(d_{\text{col}}^2)$ that is used in the PTQ baselines (Frantar et al. 2023; Chee et al. 2023). We then aggregate the row-wise Hessians to further reduce the required memory as described in the next section.

Aggregation of Row-wise Hessians

Computation of the row-wise Hessians, $\bar{\mathbf{H}}_{\text{OAC}_j}$, used in equation (7) is associated with a large memory footprint. We propose replacing $\bar{\mathbf{H}}_{\text{OAC}_j}$ with $\widehat{\mathbf{H}}_{\text{OAC}} = \sum_{j=1}^{d_{\text{row}}} \bar{\mathbf{H}}_{\text{OAC}_j}$ to mitigate the memory complexity. Thus, using equation (6) our simplified optimization problem transforms to

$$\begin{aligned} \underset{\delta\mathbf{W}}{\arg\min} \quad & \varepsilon_{\text{OAC}} \simeq \text{tr}(\delta\mathbf{W}\widehat{\mathbf{H}}_{\text{OAC}}\delta\mathbf{W}^\top), \\ \text{s.t.} \quad & \delta\mathbf{W}_{:,q} = \widehat{\mathbf{W}}_{:,q} - \mathbf{W}_{:,q}. \end{aligned} \tag{8}$$

Note that $\text{tr}(\delta\mathbf{W}\widehat{\mathbf{H}}_{\text{OAC}}\delta\mathbf{W}^\top)$ is an upper bound for $\sum_{j=1}^{d_{\text{row}}} \delta\mathbf{W}_{j,:} \bar{\mathbf{H}}_{\text{OAC}_j} \delta\mathbf{W}_{j,:}^\top$ since all $\bar{\mathbf{H}}_{\text{OAC}_j}$ matrices are positive definite. Hence, equation (8) approximates (7) as supported empirically in our experiments.

Approximation of the Output-adaptive Hessian

To further clarify our proposed output-adaptive approach, we start this section by computing the output-adaptive Hessian for the weights of a logistic regression classifier. We then generalize our approach to the linear layers of LLMs.

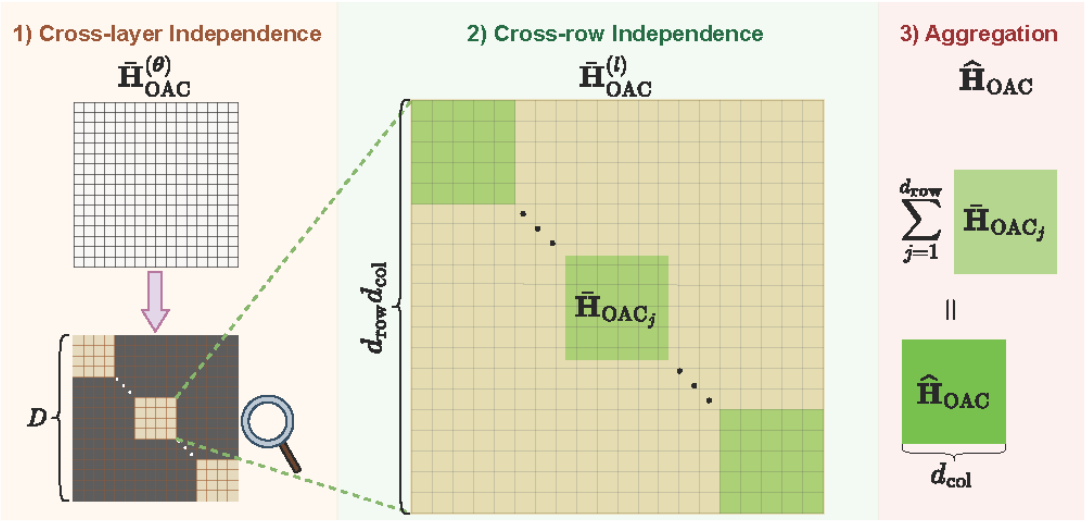

Figure 2: This figure shows our proposed steps to reduce the computational complexity of the output-adaptive Hessian. 1) The Hessian of each linear layer is independently computed. 2) The Hessian of each linear layer becomes block diagonal according to the rows independence assumption. 3) All of the row-wise Hessians are aggregated to reduce the memory footprint.

$\bar{\mathbf{H}}_{\text{OAC}}$ for a Binomial Logistic Regression Classifier Let $\mathbf{w} \in \mathbb{R}^d$ be the weights of a logistic regression model trained using cross-entropy. $\mathbf{x}_i \in \mathbb{R}^d$ is the input, and $y_i \in \{0, 1\}$ is its label. Equation (9) shows the underlying mechanism of the model. Equations (10) also shows the gradient obtained for the i^{th} sample.

$$P_{\mathbf{w}}(y_i = 1|\mathbf{x}_i) = \pi_{\mathbf{w}}(\mathbf{x}_i) = \frac{e^{\mathbf{w}^\top \mathbf{x}_i}}{1 + e^{\mathbf{w}^\top \mathbf{x}_i}} \tag{9}$$

$$\mathbf{g}[i] := \frac{\partial \mathcal{L}_{\text{CE}}(\mathbf{w}; \mathbf{x}_i, y_i)}{\partial \mathbf{w}} = \mathbf{x}_i \Big[\pi_{\mathbf{w}}(\mathbf{x}_i) - y_i\Big] \tag{10}$$

Based on the Fisher Information Identity, we approximate[1] the expected Hessian over N samples as

$$\begin{aligned}\bar{\mathbf{H}}_{\text{OAC}} &= \mathbb{E}_{(\mathbf{x}_i, y_i)}\left[\frac{\partial^2 \mathcal{L}_{\text{CE}}(\mathbf{w}; \mathbf{x}_i)}{\partial \mathbf{w} \partial \mathbf{w}^\top}\right] \\ &= \mathbb{E}_{(\mathbf{x}_i, y_i)}\left[\frac{\partial \mathcal{L}_{\text{CE}}(\mathbf{w}; \mathbf{x}_i, y_i)}{\partial \mathbf{w}} \frac{\partial \mathcal{L}_{\text{CE}}(\mathbf{w}; \mathbf{x}_i, y_i)^\top}{\partial \mathbf{w}}\right] \\ &\simeq \frac{1}{N}\sum_{i=1}^{N} \mathbf{g}[i]\mathbf{g}[i]^\top. \end{aligned} \tag{11}$$

As shown in equation (11), the term y_i appears when approximating the second derivative with $\mathbf{g}[i]\mathbf{g}[i]^\top$ based on the Fisher Information Identity. Hence, we call our method output-adaptive.

Generalization of $\widehat{\mathbf{H}}_{\text{OAC}}$ for Linear Layers of LLMs Having established the computation of $\bar{\mathbf{H}}_{\text{OAC}}$ for a binomial logistic regression classifier, we propose to generalize our approach to compute $\widehat{\mathbf{H}}_{\text{OAC}}$ for the linear layers of LLMs as follows. Let $\mathbf{W} \in \mathbb{R}^{d_{\text{row}} \times d_{\text{col}}}$ be the weight matrix of the l^{th} linear layer in an LLM resulted by concatenating d_{row} rows. Following the cross-layer independence assumption, the Hessian of $\mathbf{W}$ is independent of other layers. Also, based on the cross-row independence assumption, the Hessian for each row of $\mathbf{W}$ is independent of other rows. Using equation (11), the output-adaptive Hessian of the j^{th} row is approximated by equation (12), where $\mathbf{G}_{j,:}[i]$ is the j^{th} row of the gradient matrix, $\mathbf{G}[i] \in \mathbb{R}^{d_{\text{row}} \times d_{\text{col}}}$, computed for the i^{th} calibration sample.

$$\bar{\mathbf{H}}_{\text{OAC}_j} \simeq \frac{1}{N}\sum_{i=1}^{N} \mathbf{G}_{j,:}^\top[i]\mathbf{G}_{j,:}[i]. \tag{12}$$

Then, all of the row-wise Hessians are aggregated to compute $\widehat{\mathbf{H}}_{\text{OAC}}$. However, to optimize the computations, equation (13) is used to directly compute the aggregation of all row-wise Hessians without individually computing them

$$\begin{aligned}\widehat{\mathbf{H}}_{\text{OAC}} \simeq \sum_{j=1}^{d_{\text{row}}} \bar{\mathbf{H}}_{\text{OAC}_j} &\simeq \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{d_{\text{row}}} \mathbf{G}_{j,:}[i]^\top \mathbf{G}_{j,:}[i] \\ &= \frac{1}{N}\sum_{i=1}^{N} \mathbf{G}[i]^\top \mathbf{G}[i]. \end{aligned} \tag{13}$$

5 OAC Pipeline

Our proposed method, OAC, comprises two main phases for PTQ of LLMs weights. (i) Computing the output-adaptive Hessian of each weight matrix. (ii) Calibrating each weight

[1] We refer to (Edalati et al. 2024, Appendix A) for the proof.

Algorithm 1: OAC Pipeline

Require: model, N data samples
for $j = 1$ **to** model.num_layers **do**
  block := model.block[j]
  ## Phase 1: Computation of $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ ##
  for $i = 1$ **to** N **do**
    loss = Cross-Entropy(model(data[i]), data[i])
    loss.backward()
    for layer $\in$ block.linear_layers **do**
      $\mathbf{W}$:= layer.weight
      $\mathbf{G}[i] = \partial^{\text{loss}}/\partial \mathbf{W}$
      $\widehat{\mathbf{H}}_{\mathrm{OAC}} += \frac{1}{N}\mathbf{G}^\top[i]\mathbf{G}[i]$
    end for
  end for
  ## Phase 2: Calibration by $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ ##
  for layer $\in$ block.linear_layers **do**
    $\widehat{\mathbf{W}} =$ Hessian-based Calibration$(\mathbf{W}, \widehat{\mathbf{H}}_{\mathrm{OAC}})$
  end for
end for

matrix using its output-adaptive Hessian. To develop a complete PTQ pipeline, we integrate a Hessian-based calibration technique with our proposed method. The OAC pipeline is described in Algorithm 1.

Computation of $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ OAC uses equation (13) to approximate the Hessian of each weight matrix. Therefore, it requires computing the gradients of each weight matrix for all calibration samples. We propose computing the gradients of the linear layers inside each transformer block simultaneously, while other transformer blocks are frozen. This avoids repeating the costly output generation and backpropagation for all linear layers. As described in Algorithm 1, the transformer blocks are iteratively selected. For each block, the model outputs are generated using the calibration samples. Following the calculation of the cross-entropy loss, OAC employs backpropagation to compute the gradients of the weights associated with the linear layers within the block. Then, $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ of the linear layers within the block are separately approximated based on equation (13) and are used in the next phase of our proposed PTQ method.

Calibration by $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ Most of the Hessian-based calibration techniques can be employed in this phase. However, to apply OAC for accurate 2-bit PTQ of LLMs, the following steps from SpQR (Dettmers et al. 2024) are integrated into our method. The salient weights are detected and isolated using equation (4) while $\widehat{\mathbf{H}}$ is replaced by our proposed output-adaptive Hessian $\widehat{\mathbf{H}}_{\mathrm{OAC}}$. Then, the column-wise updating process is performed to reduce the distortion of the cross-entropy loss. Likewise, the update term is computed by replacing $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ in equation (3). Finally, the scales and zeros are quantized to reduce the average bit hence enabling smaller group quantization and keeping more outliers in the FP32 format. Furthermore, to show the effectiveness of our proposed output-adaptive Hessian computation in binary PTQ, we integrated the $\widehat{\mathbf{H}}_{\mathrm{OAC}}$ to the calibration procedure of BiLLM (Huang et al. 2024, Algorithm 1). Experimental results presented in Section 6 show that our proposed approach surpasses both BiLLM and SpQR in the PTQ of LLMs.

6 Experiments and Results

This section provides our main results as well as a summary of the experimental settings. We refer to the Appendix of (Edalati et al. 2024) for additional results, ablation studies, and details of the experimental settings.

Experimental Settings

We investigate various language models with different sizes including OPT (Zhang et al. 2022), LLaMa (Touvron et al. 2023a), and LLaMa 2 (Touvron et al. 2023b) families. The calibration set comprises 128 sequences of 2048 tokens. To evaluate the performance of the quantized models on language modeling tasks, we report their *perplexity* on C4 (Raffel et al. 2020) and WikiText2 (Merity et al. 2017). Also, Language Model Evaluation Harness (LMEH) (Gao et al. 2023) is utilized for evaluating the reasoning abilities of the quantized models. We report the zero-shot accuracy on WinoGrande (Sakaguchi et al. 2021), PiQA (Tata and Patel 2003), HellaSwag (Zellers et al. 2019), ARC-easy, and ARC-challenge (Clark et al. 2018) in addition to the five-shot exact match on GSM8K (Cobbe et al. 2021) datasets.

In our experimental results, we compare our method with the latest state-of-the-art PTQ methods that are developed for LLMs including Round to Nearest (RTN) (Dettmers et al. 2022), OPTQ (Frantar et al. 2023), QuIP (Chee et al. 2023), and SpQR (Dettmers et al. 2024) on 2- and 3- bit quantization[2] in addition to OmniQuant (Shao et al. 2024) as an efficient QAT method. SqueezeLLM (Kim et al. 2024) is also included in the 3-bit quantization experiments. Moreover, BiLLM (Huang et al. 2024) is selected as the most recent baseline for binary PTQ.

Results

Quantization of LLMs becomes more complicated and exhibits a larger accuracy degradation as (i) the model size decreases, and (ii) the overall average bit width to accommodate lower precision weights, fewer mitigated outliers, and larger group quantization decreases. As such, we investigate 2-bit PTQ of small to large LLMs with 1.3B to 30B parameters. Table 1 compares the performance of 2-bit quantized LLaMa and OPT models on the language modeling and reasoning tasks. Our experimental results show that OAC significantly outperforms the state-of-the-art baselines.

To further evaluate our proposed method, OAC, at extreme low-precision quantization, we apply OAC to the binarization of the LLaMa family. Table 2 shows the performance of OAC compared to BiLLM. Our experimental results show that OAC significantly outperforms BiLLM.

Our experimental results reveal the importance of output-adaptive calibration in low-precision quantization scenarios. Our proposed approach, OAC, performs better than other

[2]See (Edalati et al. 2024, Appendix H) for detailed binary, 2-, and 3-bit PTQ results.

Method	Avg Bits	C4	WikiText2	LMEH	Method	Avg Bits	C4	WikiText2	LMEH
LLaMa2-7B					LLaMa2-13B				
Baseline	16	7.03	5.47	56.19	Baseline	16	6.50	4.88	60.34
RTN	2.25	4.6e3	4.3e3	29.37	RTN	2.25	1.4e2	1.2e2	32.36
OPTQ	2.25	1.5e2	2.0e2	30.46	OPTQ	2.25	52.34	47.09	31.81
OmniQuant	2.25	15.74	11.16	39.34	OmniQuant	2.25	11.62	8.31	44.95
QuIP	2	51.22	67.92	32.06	QuIP	2	11.63	9.81	44.78
SpQR	2.09	13.22	11.09	42.90	SpQR	2.09	9.81	7.58	49.29
OAC (ours)	2.09	**11.90**	**9.48**	**45.56**	OAC (ours)	2.09	**9.49**	**7.39**	**49.96**
LLaMa-13B					LLaMa-30B				
Baseline	16	6.61	5.09	58.86	Baseline	16	5.95	4.10	64.39
RTN	2.25	4.5e2	7.8e2	31.40	RTN	2.25	99.23	68.06	33.23
OPTQ	2.25	24.56	19.16	30.88	OPTQ	2.25	11.07	8.63	46.29
OmniQuant	2.25	10.70	7.78	47.03	OmniQuant	2.25	9.69	6.98	49.04
QuIP	2	10.95	9.30	46.11	QuIP	2	9.28	7.48	49.51
SpQR	2.09	9.61	7.63	48.35	SpQR	2.09	8.25	**6.29**	54.16
OAC (ours)	2.09	**9.45**	**7.45**	**50.13**	OAC (ours)	2.09	**8.22**	6.31	**54.97**
OPT-13B					OPT-30B				
Baseline	16	11.54	10.13	58.72	Baseline	16	10.91	9.56	60.97
RTN	2.25	2.7e4	7.6e4	34.85	RTN	2.25	6.4e3	1.3e4	35.12
OPTQ	2.25	15.67	15.62	50.14	OPTQ	2.25	13.43	12.85	54.05
OmniQuant	2.25	20.47	15.70	50.71	OmniQuant	2.25	13.65	11.38	55.59
QuIP	2	14.07	13.41	53.76	QuIP	2	12.41	11.27	57.20
SpQR	2.09	13.28	12.16	55.07	SpQR	2.09	12.00	10.72	57.63
OAC (ours)	2.10	**13.25**	**11.75**	**55.86**	OAC (ours)	2.09	**11.99**	**10.48**	**58.20**

Table 1: Comparison of the perplexity and reasoning score for 2-bit quantized LLaMa and OPT models. The "LMEH" column shows the average score over the LMEH reasoning tasks.

Method	Avg Bits	C4	WikiText2	LMEH	Method	Avg Bits	C4	WikiText2	LMEH
LLaMa2-7B					LLaMa2-13B				
Baseline	16	7.03	5.47	56.19	Baseline	16	6.50	4.88	60.34
BiLLM	1.08	28.00	25.59	35.61	BiLLM	1.08	23.06	18.54	37.01
OAC (ours)	1.09	**21.64**	**19.50**	**38.74**	OAC (ours)	1.08	**15.25**	**13.15**	**42.42**
LLaMa-7B					LLaMa-13B				
Baseline	16	7.11	5.68	55.15	Baseline	16	6.61	5.09	58.86
BiLLM	1.09	27.82	30.73	35.88	BiLLM	1.09	**14.93**	14.13	42.65
OAC (ours)	1.09	**19.82**	**17.79**	**38.84**	OAC (ours)	1.09	15.17	**14.05**	**44.31**

Table 2: Comparison of the perplexity and reasoning score for binarized LLaMa models. The "LMEH" column shows the average score over the LMEH reasoning tasks.

methods, especially, in more complicated scenarios where the average bit width or model size is relatively smaller.

7 Conclusion

We proposed a novel Output-adaptive Calibration (OAC) approach to incorporate the model output in PTQ of LLMs. Our proposed method, OAC, minimizes the quantization error based on the distortion of the output cross-entropy loss, leading to a more accurate quantized model. To reduce the computational complexity of OAC, we introduced an approximation to the output-adaptive Hessian based on the Fisher Information Identity. We also delved into the details of deploying our approximated Hessian to update the weight matrices and identify the salient weights within the PTQ pipeline. We compared OAC with several PTQ methods on various LLMs, and found OAC to be more accurate on various language modeling and reasoning tasks. Our experimental results demonstrate that our proposed method, OAC, outperforms the state-of-the-art by a significant margin in extreme low-precision (e.g. 2-bit and binary) PTQ.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Babenko, A.; and Lempitsky, V. 2014. Additive Quantization for Extreme Vector Compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Chee, J.; Cai, Y.; Kuleshov, V.; and De Sa, C. M. 2023. QuIP: 2-Bit Quantization of Large Language Models With Guarantees. In Oh, A.; Neumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 4396–4429. Curran Associates, Inc.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *ArXiv*, abs/1803.05457.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *ArXiv*, abs/2110.14168.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. GPT3.int8(): 8-bit Matrix Multiplication for Transformers at Scale. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 30318–30332. Curran Associates, Inc.

Dettmers, T.; Svirschevski, R. A.; Egiazarian, V.; Kuznedelev, D.; Frantar, E.; Ashkboos, S.; Borzunov, A.; Hoefler, T.; and Alistarh, D. 2024. SpQR: A Sparse-Quantized Representation for Near-Lossless LLM Weight Compression. In *The Twelfth International Conference on Learning Representations*.

Du, D.; Zhang, Y.; Cao, S.; Guo, J.; Cao, T.; Chu, X.; and Xu, N. 2024. BitDistiller: Unleashing the Potential of Sub-4-Bit LLMs via Self-Distillation. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 102–116. Bangkok, Thailand: Association for Computational Linguistics.

Edalati, A.; Ghaffari, A.; Asgharian, M.; Hou, L.; Chen, B.; and Nia, V. P. 2024. OAC: Output-adaptive Calibration for Accurate Post-training Quantization. *arXiv preprint arXiv:2405.15025*.

Egiazarian, V.; Panferov, A.; Kuznedelev, D.; Frantar, E.; Babenko, A.; and Alistarh, D. 2024. Extreme Compression of Large Language Models via Additive Quantization. In Salakhutdinov, R.; Kolter, Z.; Heller, K.; Weller, A.; Oliver, N.; Scarlett, J.; and Berkenkamp, F., eds., *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 12284–12303. PMLR.

Frantar, E.; and Alistarh, D. 2022. Optimal Brain Compression: A Framework for Accurate Post-Training Quantization and Pruning. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 4475–4488. Curran Associates, Inc.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2023. OPTQ: Accurate Quantization for Generative Pre-trained Transformers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

Gao, L.; Tow, J.; Abbasi, B.; Biderman, S.; Black, S.; DiPofi, A.; Foster, C.; Golding, L.; Hsu, J.; Le Noac'h, A.; Li, H.; McDonell, K.; Muennighoff, N.; Ociepa, C.; Phang, J.; Reynolds, L.; Schoelkopf, H.; Skowron, A.; Sutawika, L.; Tang, E.; Thite, A.; Wang, B.; Wang, K.; and Zou, A. 2023. A framework for few-shot language model evaluation.

Gholami, A.; Kim, S.; Dong, Z.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2022. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, 291–326. Chapman and Hall/CRC.

Halko, N.; Martinsson, P. G.; and Tropp, J. A. 2011. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2): 217–288.

Hassibi, B.; and Stork, D. 1992. Second order derivatives for network pruning: Optimal Brain Surgeon. In Hanson, S.; Cowan, J.; and Giles, C., eds., *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann.

Huang, W.; Liu, Y.; Qin, H.; Li, Y.; Zhang, S.; Liu, X.; Magno, M.; and Qi, X. 2024. BiLLM: Pushing the Limit of Post-Training Quantization for LLMs. In Salakhutdinov, R.; Kolter, Z.; Heller, K.; Weller, A.; Oliver, N.; Scarlett, J.; and Berkenkamp, F., eds., *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 20023–20042. PMLR.

Hubara, I.; Nahshan, Y.; Hanani, Y.; Banner, R.; and Soudry, D. 2021. Accurate Post Training Quantization With Small Calibration Sets. In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 4466–4475. PMLR.

Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; and Kalenichenko, D. 2018. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kim, J.; Lee, J. H.; Kim, S.; Park, J.; Yoo, K. M.; Kwon, S. J.; and Lee, D. 2023. Memory-Efficient Fine-Tuning of Compressed Large Language Models via sub-4-bit Integer Quantization. In Oh, A.; Neumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in*

Neural Information Processing Systems, volume 36, 36187–36207. Curran Associates, Inc.

Kim, S.; Hooper, C. R. C.; Gholami, A.; Dong, Z.; Li, X.; Shen, S.; Mahoney, M. W.; and Keutzer, K. 2024. SqueezeLLM: Dense-and-Sparse Quantization. In Salakhutdinov, R.; Kolter, Z.; Heller, K.; Weller, A.; Oliver, N.; Scarlett, J.; and Berkenkamp, F., eds., *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 23901–23923. PMLR.

LeCun, Y.; Denker, J.; and Solla, S. 1989. Optimal Brain Damage. In Touretzky, D., ed., *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann.

Li, H.; De, S.; Xu, Z.; Studer, C.; Samet, H.; and Goldstein, T. 2017. Training Quantized Nets: A Deeper Understanding. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Li, Y.; Gong, R.; Tan, X.; Yang, Y.; Hu, P.; Zhang, Q.; Yu, F.; Wang, W.; and Gu, S. 2021. BRECQ: Pushing the Limit of Post-Training Quantization by Block Reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Chen, W.-M.; Wang, W.-C.; Xiao, G.; Dang, X.; Gan, C.; and Han, S. 2024. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. In Gibbons, P.; Pekhimenko, G.; and Sa, C. D., eds., *Proceedings of Machine Learning and Systems*, volume 6, 87–100.

Liu, Z.; Oguz, B.; Zhao, C.; Chang, E.; Stock, P.; Mehdad, Y.; Shi, Y.; Krishnamoorthi, R.; and Chandra, V. 2024. LLM-QAT: Data-Free Quantization Aware Training for Large Language Models. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 467–484. Bangkok, Thailand: Association for Computational Linguistics.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2017. Pointer Sentinel Mixture Models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Nagel, M.; Amjad, R. A.; Van Baalen, M.; Louizos, C.; and Blankevoort, T. 2020. Up or Down? Adaptive Rounding for Post-Training Quantization. In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 7197–7206. PMLR.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140): 1–67.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. WinoGrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9): 99–106.

Shao, W.; Chen, M.; Zhang, Z.; Xu, P.; Zhao, L.; Li, Z.; Zhang, K.; Gao, P.; Qiao, Y.; and Luo, P. 2024. OmniQuant: Omnidirectionally Calibrated Quantization for Large Language Models. In *The Twelfth International Conference on Learning Representations*.

Tata, S.; and Patel, J. 2003. PiQA: an algebra for querying protein data sets. In *15th International Conference on Scientific and Statistical Database Management, 2003.*, 141–150.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023a. Llama: Open and efficient foundation language models (2023). *arXiv preprint arXiv:2302.13971*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tseng, A.; Chee, J.; Sun, Q.; Kuleshov, V.; and De Sa, C. 2024. QuIP#: Even Better LLM Quantization with Hadamard Incoherence and Lattice Codebooks. In Salakhutdinov, R.; Kolter, Z.; Heller, K.; Weller, A.; Oliver, N.; Scarlett, J.; and Berkenkamp, F., eds., *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 48630–48656. PMLR.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. In Krause, A.; Brunskill, E.; Cho, K.; Engelhardt, B.; Sabato, S.; and Scarlett, J., eds., *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, 38087–38099. PMLR.

Yao, Z.; Yazdani Aminabadi, R.; Zhang, M.; Wu, X.; Li, C.; and He, Y. 2022. ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 27168–27183. Curran Associates, Inc.

Yuan, Z.; Shang, Y.; and Dong, Z. 2024. PB-LLM: Partially Binarized Large Language Models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In Korhonen, A.; Traum, D.; and Màrquez, L., eds., *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4791–4800. Florence, Italy: Association for Computational Linguistics.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.