LOW-RANK CORRECTION FOR QUANTIZED LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of model compression for Large Language Models (LLMs) at post-training time, where the task is to compress a well-trained model using only a small set of calibration input data. In this work, we introduce a new low-rank approach to correct for quantization errors of both weights and activations in LLMs: we propose to add low-rank weight matrices in full precision that act on the *unquantized* activations. We then solve a joint optimization problem over the quantized representation of the weights and additional low-rank weight matrices. We focus on the case of 4-bit weight-and-activation quantization (W4A4). Using ranks equivalent to 10% of the original weight matrix size, our approach reduces the accuracy gap with the original model by more than 50%. Using ranks equivalent to 30% of the original weight matrix, the accuracy gap is closed completely. We demonstrate our results on three recent LLMs, namely Llama-3, Phi-3 and Mixtral models.

1 INTRODUCTION

Large Language Models (LLMs) (Abdin et al., 2024; Dubey et al., 2024; Jiang et al., 2024) have demonstrated exceptional performances across a wide range of applications. However, due to their massive size, these models require considerable computational and memory resources at inference.

Post-training quantization (PTQ) is among the most important techniques to solve both memory and compute issues during LLM inference. The majority of quantization schemes focus on compressing LLMs by using weight-only quantization (Frantar et al., 2022; Shao et al., 2023; Dettmers et al., 2023). One major limitation of PTQ is the presence of magnitude outliers in the model layer weights, which can severely affects the quantization process (Wei et al., 2022; Xiao et al., 2023) and deteriorate the performances of quantized models. To handle them, several offline approaches has been proposed in the literature, such as mixed-precision strategies (Dettmers et al., 2023), adapted rescaling Lin et al. (2024), and incoherence processing Chee et al. (2024); Tseng et al. (2024). Recently, several works proposed to use additional low-rank weight matrices in full precision to correct for quantization errors in the weights Kang et al. (2024); Ou et al. (2024).

Weight only quantization methods enable to store LLMs on smaller devices, and accelerate the General Matrix-Vector Multiply (GEMV) operators in the decoding stage Lin et al. (2024); Frantar et al. (2022), however, these approaches still require to keep activations in full precision (usually FP16). To improve on this, several works (Ashkboos et al., 2023; Xiao et al., 2023; Dettmers et al., 2022; Zhao et al., 2024) aim at jointly quantizing the weights and activations (and sometime KV cache) to compute the forward pass in low bit precision. Unlike weight quantization, the quantization of activations requires online strategies to compute their low bit representations on the fly (Jacob et al., 2018).To deal with outliers in activations, (Xiao et al., 2023) propose to scale the weights, thus reducing the magnitude range of activations. (Ashkboos et al., 2024; Liu et al., 2024) propose to process weights and activations using Hadamard transform. More recently Zhang et al. (2024) propose to add low-rank correction terms in full precision in order to correct for quantization errors. Although these approaches has been shown to be effective at W4A8, they still struggle to deal with the case where both weights and activations are quantized in lower bit precision such as W4A4, leaving an opportunity to improve on current methods in these harder settings.

Contributions. In this work, we improve on current SoTA approaches for PTQ, and introduce LRC, a new method that leverages low-rank weight matrices in full precision to correct for both weight and activation quantization errors. By jointly optimizing for quantized representations of the original weights and full precision low-rank weights matrices, our method allows for quantizing weights, and activations (and KV caches) to 4 bits with minimal loss in accuracy. Our main contributions are summarized below.

- We introduce a general framework that aims at jointly optimizing for quantized representations of the original weights acting on the quantized activations, as well as full precision low-rank weights matrices operating on the *unquantized* activations.
- We derive an alternative scheme to solve the joint optimization problem and obtain a simple algorithm easily compatible with recent quantization techniques, such as QuaRot (Ashkboos et al., 2024) and GPTQ (Frantar et al., 2022).
- Using our quantization scheme, LRC manages to reduce the accuracy gap with original models by more than 50% using only low-rank matrices with 10% of the original size, and outperform all current approaches at W4A4.

1.1 RELATED WORK

Dealing with Outliers. One major limitation of quantization techniques is the presence of outliers in both weights and activations that can deteriorate the quality of the approximations, and lead to a considerable drop in performance. A recent line of works (Ashkboos et al., 2024; Liu et al., 2024) proposed to apply specific rotations (and their inverses), on both weights and activations in order to remove outliers while preserving exactly the output of the original model. Such a pre-processing step, in combination with the GPTQ Algorithm (Frantar et al., 2022) enabling efficient quantization of the weights, and lead to SoTA performances in quantization at W4A8. In this work, we leverage this pre-processing step, and consider these methods as baselines for our approach where low-rank weight matrices in full precisions are added to the forward pass in order to correct for quantization errors on both weights and activations, enabling to quantize LLMs at W4A4 with a 50% gain.

Low-Rank Correction. Recent works proposed to leverage low-rank matrices to reduce quantization errors in LLMs (Zhang et al., 2024; Ou et al., 2024). In (Zhang et al., 2024), the authors consider the case where both weights and activations are quantized, and propose to add well-chosen low-rank matrices in full precision in the forward pass to reduce the quantization errors. To deal with outliers, they first compute some rescaling matrices (Lin et al., 2024; Xiao et al., 2023) using a calibration dataset, and then propose to add low-rank approximation of the rescaled residual errors between the original and quantized weights obtained by SVD. In (Ou et al., 2024), the authors improve on the low-rank approach introduced in Zhang et al. (2024) and consider to apply a PCA on the output activation errors using a calibration dataset. However, their work only focus on the quantization of the weights, leaving aside the quantization of activations. In this work, we improve on prior research by incorporating both the empirical distribution of activations and the errors induced by activation quantization into our analysis to optimize the low-rank weight matrices.

2 BACKGROUND ON POST-TRAINING QUANTIZATION

Weight Quantization. Weight quantization aims at obtaining new weights with a lower bit precision, reducing the memory needed to store the model, while preserving the output of the original model. One standard approach is to perform layer-wise quantization, where quantized weights are obtained by solving a reconstruction problem on a calibration dataset. Given a small dataset of n sampled activations $\boldsymbol{X}_\ell \in \mathbb{R}^{d_\ell^{\text{in}} \times n}$ at a certain layer ℓ, and the associated weight matrix $\boldsymbol{W}_\ell \in \mathbb{R}^{d_\ell^{\text{out}} \times d_\ell^{\text{in}}}$, the goal is to find a matrix of quantized weights $\widehat{\boldsymbol{W}}_\ell$ which minimizes the quadratic error over the dataset (Nagel et al., 2020):

$$\min_{\widehat{\boldsymbol{W}}_\ell \in \mathcal{C}(b) \cap \mathbb{R}^{d_\ell^{\text{out}} \times d_\ell^{\text{in}}}} \mathcal{L}_{\text{q}}(\widehat{\boldsymbol{W}}_\ell) := \|\boldsymbol{W}_\ell \boldsymbol{X}_\ell - \widehat{\boldsymbol{W}}_\ell \boldsymbol{X}_\ell\|_2^2 \,, \tag{1}$$

where $\mathcal{C}(b)$ is the constraint set of matrices admitting a certain bit per weight $b > 0$ precision. Due to the non-convexity of the constraints, finding the exact solution of the problem is hard, and many works have focused on designing algorithms to efficiently approximate a solution (Frantar et al., 2022; van Baalen et al., 2024; Lin et al., 2024; Egiazarian et al., 2024).

Activation Quantization. While weight quantization enables to store large models at a lower memory cost, it still requires additional memory space at inference time to perform the forward operations in full precision, usually by de-quantizing weights to FP16. To reduce the memory footprint and FLOP cost further, it is desirable to quantize the activations during inference to compute matrix multiplications in lower precision, usually using specialized cuda kernels that perform GEMM in low bit precision (Wei et al., 2022; Yao et al., 2022; Wu et al., 2023; Xiao et al., 2023). The quantization step of activations generally happens at every layer of a model as one might want to preserve full precision activations to perform coordinate-wise non-linear transformations of the activations (Ashkboos et al., 2024; Xiao et al., 2023). Additionally, as quantizing activations require additional operations in the forward, this step must only incur a small overhead in time and memory. Due to this constraint, previous work consider simple, yet efficient techniques such as round-to-nearest (RTN) (Ashkboos et al., 2024), to quantize activations on the fly at inference time.

In our work we assume a simple on-the-fly quantization, rescaling each activation $\boldsymbol{x}$ by $c \cdot \max(\text{abs}(\boldsymbol{x}))$ and rounding to the nearest integer. We perform a simple hyper-parameter search for c. Our main focus is deriving a low-rank correction to the weight matrix that accounts for some of the error incurred in activation quantization.

3 LOW-RANK CORRECTION

In this work, we propose to add full precision low-rank weight matrices in the forward pass that act on the *unquantized* activations to correct for quantization errors of both weights and activations. In the following, we start by presenting the proposed framework to quantize a single weight matrix while correcting for quantization errors using additional low-rank weight matrices. Then we introduce our proposed algorithm that effectively computes the quantized representations of the original weights in low-bit precision and the low-rank weight matrices in full precision. We conclude this section by providing a detailed overview of the approach when applied on a standard LLM.

3.1 GENERAL FRAMEWORK

Before introducing our framework, we need to establish some clarifying notations.

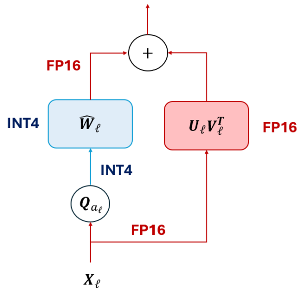

Figure 1: Computational scheme of our method, where both weights and activations are quantized, and a low-rank matrix in full precision is added and operates on the *unquantized* activations.

Notations. Given a calibration dataset $\boldsymbol{X} \in \{1, ..., D\}^n$ of token ids where D is the dictionary size, and a model $\mathcal{M}$ with L layers, we denote $(\boldsymbol{X}_\ell)_{1\leq\ell\leq L}$ where $\boldsymbol{X}_\ell \in \mathbb{R}^{d_\ell^{\text{in}} \times n}$, the sequence of activations obtained along the forward pass of $\mathcal{M}$ at each layer ℓ when applied on $\boldsymbol{X}$. We also denote $(\boldsymbol{W}_\ell)_{1\leq\ell\leq L}$ where $\boldsymbol{W}_\ell \in \mathbb{R}^{d_\ell^{\text{out}} \times d_\ell^{\text{in}}}$, the sequence of weight matrices of $\mathcal{M}$ which act on the activations $(\boldsymbol{X}_\ell)_{1\leq\ell\leq L}$ respectively along the forward pass. In the following, we always consider the case where $n \geq \max\limits_{1\leq\ell\leq L}(d_\ell^{\text{out}}, d_\ell^{\text{in}})$, as in practice we use 128 sentences of 2048 tokens, giving $n \simeq 200k$ while in general $d \simeq 10k$. We also denote for $b > 0$, $\mathcal{C}(b)$ the constraint set of matrices admitting a certain bit per weight $b > 0$ precision, and $Q_b(\cdot)$ any quantization operator, that given an input matrix $\boldsymbol{Z} \in \mathbb{R}^{d\times d'}$ produces a matrix of same shape and satisfying $Q_b(\boldsymbol{Z}) \in \mathcal{C}(b)$. Finally, we call two optimization problems equivalent if they have the same solutions.

Low-Rank Correction Problem. We propose to extend the framework introduced in equation 1 for layer-wise quantization of weight matrices only, by also accounting for the errors induced by the quantization of activations and by adding low-rank weight matrices to correct for quantization errors of both weights and activations. Given the original activations $\boldsymbol{X}_\ell$ obtained at layer ℓ, the associated weight matrix $\boldsymbol{W}_\ell$, a bit precision for the weights $b_\ell > 0$, a bit precision for the activations $a_\ell > 0$, a quantization operator $Q_{a_\ell}(\cdot)$ acting on the activations at inference time, and a rank $1 \leq k_\ell \leq$

Algorithm 1 Update-Quant$(\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{Y}\boldsymbol{Y}^\top, \boldsymbol{X}\boldsymbol{Y}^\top, b)$

```
1: Input: Original weight matrix $\boldsymbol{W}$, low-rank weight matrices $\boldsymbol{U}, \boldsymbol{V}$, covariance matrix $\boldsymbol{Y}\boldsymbol{Y}^\top$, cross-covariance matrix $\boldsymbol{X}\boldsymbol{Y}^\top$, and the bit precision $b$.
2: Output: Quantized weight matrix $\widehat{\boldsymbol{W}}$.
3: $\boldsymbol{L}_{\boldsymbol{Y}} \leftarrow$ Cholesky$(\boldsymbol{Y}\boldsymbol{Y}^\top)$
4: $\widehat{\boldsymbol{W}} \leftarrow (\boldsymbol{W} - \boldsymbol{U}\boldsymbol{V}^\top)\boldsymbol{X}\boldsymbol{Y}^\top(\boldsymbol{L}_{\boldsymbol{Y}}^\top)^{-1}\boldsymbol{L}_{\boldsymbol{Y}}^{-1}$
5: $\widehat{\boldsymbol{W}} \leftarrow$ GPTQ$(\widehat{\boldsymbol{W}}, \boldsymbol{Y}\boldsymbol{Y}^\top, b)$ using Alg. of Frantar et al. (2022)
6: return $\widehat{\boldsymbol{W}}$
```

$\min(d_\ell^{\text{in}}, d_\ell^{\text{out}})$, we propose to consider the following reconstruction problem:

$$\min_{\substack{\widehat{\boldsymbol{W}}_\ell \in \mathcal{C}(b_\ell) \cap \mathbb{R}^{d_\ell^{\text{out}} \times d_\ell^{\text{in}}}, \\ \boldsymbol{U}_\ell \in \mathbb{R}^{d_\ell^{\text{out}} \times k_\ell}, \boldsymbol{V}_\ell \in \mathbb{R}^{d_\ell^{\text{in}} \times k_\ell}}} \mathcal{L}_{\text{qlr}}(\widehat{\boldsymbol{W}}_\ell, \boldsymbol{U}_\ell, \boldsymbol{V}_\ell) := \|\boldsymbol{W}_\ell \boldsymbol{X}_\ell - \widehat{\boldsymbol{W}}_\ell Q_{a_\ell}(\boldsymbol{X}_\ell) - \boldsymbol{U}_\ell \boldsymbol{V}_\ell^\top \boldsymbol{X}_\ell\|_2^2 \,. \tag{2}$$

Our goal here is to jointly optimize for both a quantized weight matrix $\widehat{\boldsymbol{W}}_\ell$ and full precision low-rank weight matrices $\boldsymbol{U}_\ell$, $\boldsymbol{V}_\ell$ in order to reconstruct the original output of the weight matrix $\boldsymbol{W}_\ell$.

Comparison between $\mathcal{L}_{\text{qlr}}$ and $\mathcal{L}_{\text{q}}$. While similar in nature with the optimization problem introduced in equation 1, the reconstruction problem defined in equation 2 presents two major differences: (i) we propose to account for the quantization errors of activations by considering the quantized activation $Q_{a_\ell}(\boldsymbol{X}_\ell)$ as input to the quantized weight $\widehat{\boldsymbol{W}}_\ell$, rather than the original activation $\boldsymbol{X}_\ell$. (ii) Additionally, we propose to correct the quantization errors of weights and activations using a low-rank correction matrix $\boldsymbol{U}_\ell \boldsymbol{V}_\ell^\top$ in full precision and applied on the *unquantized* activations $\boldsymbol{X}_\ell$. Figure 1 illustrates the computational scheme proposed in this work.

3.2 LRC ALGORITHM

Let us now present the proposed algorithm to solve equation 2. In the following we drop the dependency on ℓ of our notations for better readability. Starting from initial low-rank weight matrices $\boldsymbol{U}^{(0)}$, $\boldsymbol{V}^{(0)}$, we propose to alternatively optimize for $\widehat{\boldsymbol{W}}$ and $(\boldsymbol{U}, \boldsymbol{V})$ according to $\mathcal{L}_{\text{qlr}}$. For $t = 1, \dots, T$, we propose to the following updates

$$\widehat{\boldsymbol{W}}^{(t)} := \operatorname*{arg\,min}_{\widehat{\boldsymbol{W}} \in \mathcal{C}(b) \cap \mathbb{R}^{d^{\text{out}} \times d^{\text{in}}}} \|\boldsymbol{W}\boldsymbol{X} - \widehat{\boldsymbol{W}} Q_a(\boldsymbol{X}) - \boldsymbol{U}^{(t-1)}(\boldsymbol{V}^{(t-1)})^\top \boldsymbol{X}\|_2^2 \tag{3}$$

$$\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)} := \operatorname*{arg\,min}_{\boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}, \boldsymbol{V} \in \mathbb{R}^{d^{\text{in}} \times k}} \|\boldsymbol{W}\boldsymbol{X} - \widehat{\boldsymbol{W}}^{(t)} Q_a(\boldsymbol{X}) - \boldsymbol{U}\boldsymbol{V}^\top \boldsymbol{X}\|_2^2 \,. \tag{4}$$

On the Update of $\widehat{\boldsymbol{W}}$. To update $\widehat{\boldsymbol{W}}$, we rely on already existing solvers addressing equation 1. We show in the next Proposition that the optimization problem defined in equation 3 can be equivalently reformulated as a standard layer-wise quantization problem as defined in equation 1.

Proposition 3.1. *Let us denote $\boldsymbol{Y} := Q_a(\boldsymbol{X}) \in \mathcal{C}(a) \cap \mathbb{R}^{d^{in} \times n}$, and assume $\boldsymbol{Y}$ is full rank. Then, by denoting $\widetilde{\boldsymbol{W}}^{(t)} := (\boldsymbol{W} - \boldsymbol{U}^{(t)}(\boldsymbol{V}^{(t)})^\top)\boldsymbol{X}\boldsymbol{Y}^\top(\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1}$, we have that the optimization problem defined in equation 3 is equivalent to the following reconstruction problem:*

$$\min_{\widehat{\boldsymbol{W}} \in \mathcal{C}(b) \cap \mathbb{R}^{d^{out} \times d^{in}}} \|\widetilde{\boldsymbol{W}}^{(t)} \boldsymbol{Y} - \widehat{\boldsymbol{W}} \boldsymbol{Y}\|_2^2 \,. \tag{5}$$

Therefore, updating $\widehat{\boldsymbol{W}}$ according to equation 3, is equivalent to solving equation 5, which can be approximated by using any solvers designed for equation 1 such as (Frantar et al., 2022; Lin et al., 2024; Egiazarian et al., 2024). In practice, we use the GPTQ algorithm (Frantar et al., 2022) that only requires access to the target weight matrix $\widetilde{\boldsymbol{W}}^{(t)}$ and the covariance matrix $\boldsymbol{Y}\boldsymbol{Y}^\top$. Note also that in order to compute $\widetilde{\boldsymbol{W}}^{(t)}$, we need an access to the original weight matrix $\boldsymbol{W}$, the cross-covariance $\boldsymbol{X}\boldsymbol{Y}^\top$ and the current low-rank matrices $\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}$. Algorithm 1 summarizes this step.

Algorithm 2 Update-LR$(\boldsymbol{W}, \widehat{\boldsymbol{W}}, \boldsymbol{X}\boldsymbol{X}^\top, \boldsymbol{X}\boldsymbol{Y}^\top, k)$

```
1: Input: Original weight matrix $W$, quantized weight matrix $\widehat{W}$, covariance matrix $XX^\top$, cross-covariance matrix $XY^\top$, and the rank $k$.
2: Output: Low-rank weight matrices $U, V$.
3: $\Sigma_1 \leftarrow WXX^\top W^\top$
4: $\Sigma_3 \leftarrow \widehat{W}YX^\top W^\top + WXY^\top \widehat{W}^\top$
5: $L_X \leftarrow \text{Cholesky}(XX^\top), \quad S \leftarrow L_X^{-1} XY^\top \widehat{W}^\top$
6: $\Sigma_2 \leftarrow S^\top S$
7: $\Sigma \leftarrow \Sigma_1 + \Sigma_2 - \Sigma_3$
8: $U \leftarrow \text{eig}_k(\Sigma), \quad V \leftarrow \left[W^\top - (XX^\top)^{-1} XY^\top \widehat{W}^\top\right] U$
9: return $U, V$
```

Remark 3.2. In line 3 of Algorithm 1, we use the Cholesky decomposition of the covariance $\boldsymbol{Y}\boldsymbol{Y}^\top$ to compute $(\boldsymbol{W} - \boldsymbol{U}\boldsymbol{V}^\top)\boldsymbol{X}\boldsymbol{Y}^\top(\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1}$ for better numerical stability.

Remark 3.3. It is important to highlight that the choice of GPTQ (Frantar et al., 2022) in line 5 of Algorithm 1 is arbitrary. Any alternative solver capable of efficiently addressing the problem formulated in equation 5 could be employed in its place.

On the Update of $\boldsymbol{U}, \boldsymbol{V}$. While solving exactly equations 1, or 5 is still an open question due to the non-convexity of the constraints, obtaining the update for $\boldsymbol{U}, \boldsymbol{V}$, that is solving equation 4, can be done in closed form, as shown in the following Proposition.

Proposition 3.4. *Assume that $\boldsymbol{X}$ is full rank and let us denote $\boldsymbol{Y} := Q_a(\boldsymbol{X})$. Then the optimization problem defined in equation 4 is equivalent to the following optimization problem:*

$$\begin{aligned} &\max_{\boldsymbol{U} \in \mathbb{R}^{d^{out} \times k} \cap \mathcal{O}, \boldsymbol{V} \in \mathbb{R}^{d^{in} \times k}} \; \mathit{Tr}(\boldsymbol{U}^\top (\boldsymbol{\Sigma_1} + \boldsymbol{\Sigma_2}^{(t)} - \boldsymbol{\Sigma_3}^{(t)})\boldsymbol{U}) \\ &\text{s.t. } \; \boldsymbol{V} = \left[\boldsymbol{W}^\top - (\boldsymbol{X}\boldsymbol{X}^\top)^{-1}\boldsymbol{X}\boldsymbol{Y}^\top (\widehat{\boldsymbol{W}}^{(t)})^\top\right] \boldsymbol{U} \; , \end{aligned} \tag{6}$$

where $\mathcal{O}$ is the set of matrices with orthnormal columns,

$$\begin{aligned} &\boldsymbol{\Sigma_1} := \boldsymbol{W}\boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{W}^\top, \; \boldsymbol{\Sigma_2}^{(t)} := \widehat{\boldsymbol{W}}^{(t)} \boldsymbol{Y}\boldsymbol{X}^\top (\boldsymbol{X}\boldsymbol{X}^\top)^{-1} \boldsymbol{X}\boldsymbol{Y}^\top (\widehat{\boldsymbol{W}}^{(t)})^\top, \; \textit{and} \\ &\boldsymbol{\Sigma_3}^{(t)} := \widehat{\boldsymbol{W}}^{(t)} \boldsymbol{Y}\boldsymbol{X}^\top \boldsymbol{W}^\top + \boldsymbol{W}\boldsymbol{X}\boldsymbol{Y}^\top (\widehat{\boldsymbol{W}}^{(t)})^\top \; . \end{aligned}$$

In addition, a solution can be obtained by defining $\boldsymbol{U}$ as the k unit eigenvectors of $\boldsymbol{\Sigma}^{(t)} := \boldsymbol{\Sigma_1} + \boldsymbol{\Sigma_2}^{(t)} - \boldsymbol{\Sigma_3}^{(t)}$ associated to its k largest eigenvalues, and $\boldsymbol{V}$ as in equation 6.

To compute the updated $\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}$, we require an access to the original weight matrix $\boldsymbol{W}$, the current quantized approximation $\widehat{\boldsymbol{W}}^{(t)}$, and the covariance and cross-covariance $\boldsymbol{X}\boldsymbol{X}^\top$ and $\boldsymbol{X}\boldsymbol{Y}^\top$ respectively. The pseudo-code of this step is detailed in Algorithm 2. Note that in line 8 of the Algorithm 2, we denote $\text{eig}_k(\cdot)$ the operator that returns the k first unit eigenvectors of a symmetric matrix ranked in the decreasing order w.r.t their eigenvalues.

Initialization. To initialize our algorithm, that is to instantiate $\boldsymbol{U}^{(0)}$ and $\boldsymbol{V}^{(0)}$, we propose to consider a relaxed formulation of the original optimization problem defined in equation 2, where we remove the constraint on $\widehat{\boldsymbol{W}}$. More formally we consider the following optimization problem:

$$\min_{\substack{\widetilde{\boldsymbol{W}} \in \mathbb{R}^{d^{\text{out}} \times d^{\text{in}}}, \\ \boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}, \boldsymbol{V} \in \mathbb{R}^{d^{\text{in}} \times k}}} \mathcal{L}_{\text{qlr}}(\widetilde{\boldsymbol{W}}, \boldsymbol{U}, \boldsymbol{V}) \; . \tag{7}$$

In fact, equation 7 can be solved in closed form as shown in the following Proposition.

Proposition 3.5. *Let us denote $\boldsymbol{Y} := Q_a(\boldsymbol{X})$ and assume that $\boldsymbol{Y}$ is full rank. Then the optimization problem defined in equation 7 is equivalent to the following optimization problem:*

$$\max_{\substack{\widetilde{\boldsymbol{W}}\in\mathbb{R}^{d^{out}\times d^{in}},\\ \boldsymbol{U}\in\mathbb{R}^{d^{out}\times k}\cap\mathcal{O},\boldsymbol{V}\in\mathbb{R}^{d^{in}\times k}}} \quad Tr(\boldsymbol{U}^\top\boldsymbol{\Sigma}_{init}\boldsymbol{U}) \tag{8}$$
$$s.t.\;\; \boldsymbol{V}=\boldsymbol{W}^\top\boldsymbol{U} \;\; and \;\; \widetilde{\boldsymbol{W}}=\left[\boldsymbol{W}-\boldsymbol{U}\boldsymbol{V}^\top\right]\boldsymbol{X}\boldsymbol{Y}^\top(\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1}\;,$$

where $\mathcal{O}$ is the set of matrices with orthnormal columns, and $\boldsymbol{\Sigma}_{init} := \boldsymbol{W}\boldsymbol{X}[I_n - \boldsymbol{Y}^\top(\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1}\boldsymbol{Y}]\boldsymbol{X}^\top\boldsymbol{W}^\top$. In addition, a solution can be obtained by defining $\boldsymbol{U}$ as the k unit eigenvectors of $\boldsymbol{\Sigma}_{init}$ corresponding to the k largest eigenvalues, and $\boldsymbol{V}$ and $\widetilde{\boldsymbol{W}}$ as in equation 8.

Therefore we can initialize $\boldsymbol{U}^{(0)}$ and $\boldsymbol{V}^{(0)}$ according to equation 7 in closed form as long as we have access to the original weight matrix $\boldsymbol{W}$, the covariance $\boldsymbol{X}\boldsymbol{X}^\top$ and the cross-covariance $\boldsymbol{X}\boldsymbol{Y}^\top$. We present the initialization step in Algorithm 3. It is worth noting that the solution $\widetilde{\boldsymbol{W}}$ obtained in equation 8 can be used to measure an oracle performance of our alternative scheme, i.e. the effect of correcting for activation quantization, assuming a perfect weight quantizer.

Algorithm 3 Init-LR($\boldsymbol{W}, \boldsymbol{X}\boldsymbol{X}^\top, \boldsymbol{Y}\boldsymbol{Y}^\top, \boldsymbol{X}\boldsymbol{Y}^\top, k$)

```
Input: Original weight matrix $\boldsymbol{W}$, covariance matrices $\boldsymbol{X}\boldsymbol{X}^\top$, $\boldsymbol{Y}\boldsymbol{Y}^\top$, cross-covariance matrix $\boldsymbol{X}\boldsymbol{Y}^\top$, and the rank $k$.
Output: Low-rank weight matrices $\boldsymbol{U}, \boldsymbol{V}$.
$\boldsymbol{\Sigma_1} \leftarrow \boldsymbol{W}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{W}^\top$
$\boldsymbol{L_Y} \leftarrow$ Cholesky($\boldsymbol{Y}\boldsymbol{Y}^\top$),   $\boldsymbol{S} \leftarrow \boldsymbol{L_Y}^{-1}\boldsymbol{Y}\boldsymbol{X}^\top\boldsymbol{W}^\top$
$\boldsymbol{\Sigma_2} \leftarrow \boldsymbol{S}^\top\boldsymbol{S}$
$\boldsymbol{\Sigma}_{\text{init}} \leftarrow \boldsymbol{\Sigma_1} - \boldsymbol{\Sigma_2}$
$\boldsymbol{U} \leftarrow \text{eig}_k(\boldsymbol{\Sigma}_{\text{init}})$,   $\boldsymbol{V} \leftarrow \boldsymbol{W}^\top\boldsymbol{U}$
return $\boldsymbol{U}, \boldsymbol{V}$
```

Remark 3.6. Observe that the update on $\widehat{\boldsymbol{W}}$ obtained in equation 5 aims at quantizing $\widetilde{\boldsymbol{W}}^{(t)}$, which can be seen as the optimal unconstrained weight matrix when the low-rank correction matrices are fixed and set to $\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}$.

Numerical Stability. Let us now discuss the assumptions made in our previous results. In the proposed updates, we either need $\boldsymbol{X}\boldsymbol{X}^\top$, or $\boldsymbol{Y}\boldsymbol{Y}^\top$ to be full rank. To avoid the case where these matrices are singular, we add a regularization term to these matrices, and consider instead:

$$\boldsymbol{\Sigma}_x := \boldsymbol{X}\boldsymbol{X}^\top + \epsilon_x \mathrm{I}_{d^{\text{in}}}, \;\text{ and }\; \boldsymbol{\Sigma}_y := \boldsymbol{Y}\boldsymbol{Y}^\top + \epsilon_y \mathrm{I}_{d^{\text{in}}}\;,$$

where I_d denotes the identity matrix of size d. In practice we set $\epsilon_x := \frac{1e-2}{d^{\text{in}}}\mathrm{Tr}(\boldsymbol{X}\boldsymbol{X}^\top)$, and similarly, $\epsilon_y := \frac{1e-2}{d^{\text{in}}}\mathrm{Tr}(\boldsymbol{Y}\boldsymbol{Y}^\top)$.

LRC Algorithm. Finally, we present the full algorithm for LRC 4 that aims at approximating a solution of equation 2. Note that in practice, we accumulate batches of activations $\boldsymbol{X}$ to avoid running out of memory, and update $\boldsymbol{\Sigma}_x, \boldsymbol{\Sigma}_y, \boldsymbol{\Sigma}_{xy}$, as defined in lines 3, 4, 5, in an online fashion before initializing $\boldsymbol{U}, \boldsymbol{V}$ in line 6.

Application of LRC on LLMs. LRC consists of two stages: (1) the model is first pre-processed according to the QuaRot procedure (Ashkboos et al., 2024), where Hadamard rotation matrices are fused with the weights to reduce the incoherence of both weights and activations, while maintaining the outputs of the original model. (2) Then, the model is quantized using the LRC algorithm 4, where both weights and activations are quantized, and optimized low-rank matrices in FP16 precision are added to the forward pass of each weight matrix.

To compute the Hessians ($\boldsymbol{\Sigma}_{xy}, \boldsymbol{\Sigma}_x, \boldsymbol{\Sigma}_y$) we follow Frantar et al. (2022) in using calibration dataset taken from Wikitext-2: 128 randomly selected sequences of length 2048. We found that computation of these matrices required 64-bit precision for numerical accuracy. By default, our subroutine for quantization follows GPTQ (Frantar et al., 2022), (we study other subroutines in the following section). LRC works sequentially through the weight matrices of the model, computing activations for each weight matrix, obtaining the covariance and cross-covariances matrices needed to apply Algorithms 4 and solving the optimization problem 2 for each before moving to the next layer.

```
Algorithm 4 LRC($\boldsymbol{W}, \boldsymbol{X}, b, a, k$)
Input: Original weight matrix $\boldsymbol{W}$, activation $\boldsymbol{X}$, the bit precision for weights $b$, the bit precision for activation $a$, the rank $k$, and number of iterations $T$.
Output: Quantized weight matrix $\widehat{\boldsymbol{W}}$, low-rank weight matrices $\boldsymbol{U}, \boldsymbol{V}$
$\boldsymbol{\Sigma}_x \leftarrow \boldsymbol{X}\boldsymbol{X}^\top + \epsilon_x \mathrm{I}_{d^{\text{in}}}$
$\boldsymbol{Y} \leftarrow Q_a(\boldsymbol{X}), \;\; \boldsymbol{\Sigma}_y \leftarrow \boldsymbol{Y}\boldsymbol{Y}^\top + \epsilon_y \mathrm{I}_{d^{\text{in}}}$
$\boldsymbol{\Sigma}_{xy} \leftarrow \boldsymbol{X}\boldsymbol{Y}^\top$
$\boldsymbol{U}, \boldsymbol{V} \leftarrow$ Init-LR($\boldsymbol{W}, \boldsymbol{\Sigma}_x, \boldsymbol{\Sigma}_y, \boldsymbol{\Sigma}_{xy}, k$),    using Alg. 3
for $t = 1, ..., T$ do
    $\widehat{\boldsymbol{W}} \leftarrow$ Update-Quant($\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{\Sigma}_y, \boldsymbol{\Sigma}_{xy}, b$),    using Alg. 1
    $\boldsymbol{U}, \boldsymbol{V} \leftarrow$ Update-LR($\boldsymbol{W}, \widehat{\boldsymbol{W}}, \boldsymbol{\Sigma}_x, \boldsymbol{\Sigma}_{xy}, k$),    using Alg. 2
end for
return $\widehat{\boldsymbol{W}}, \boldsymbol{U}, \boldsymbol{V}$
```

4 EXPERIMENTS

We build on top of the QuaRot (Ashkboos et al., 2024) codebase, extending the method to include our approach. All our experiments focus on 4-bit quantization. We experiment with quantization of Phi-3 (mini-4k-instruct), Llama-3 (8B)and Mixtral (8x7B). All results in the table are simulated using pytorch: we leave implementation of effective kernels to other works.

4.1 BENCHMARK

We first present our main results. Our ambition is to close the gap between our main benchmark, QuaRot, and the original FP16 model by adding ranks equivalent to 10% of the original weight matrix size. To show the effect of the LRC algorithm relative to previous approaches (Zhang et al., 2024; Ou et al., 2024) we also consider a baseline of the QuaRot approach with SVD applied to the weight-matrix error (we denote this approach as SVD in tables 1 and 2).

We apply our LRC approach with a single iteration (denoted LRC (1)) and 5 iterations (LRC (5)). The runtime of our approach is comparable to the QuaRot method, though we require additional memory to store the statisitcs of the activations. Quantizing the Mixtral model on 4xA100 GPUs required 7 hours to complete 1 iteration, or 9 hours to complete 5. In general we see only modest accuracy improvements when running LRC for more iterations.

Table 1 shows the wikitext-2 perplexity (PPL) and lm-eval (Gao et al., 2024) results for each method, on each model. The tasks we considered are PIQA (PQ), HellaSwag (HS), Arc-Easy (A-e), Arc-challenge (A-c), Winogrande (WG) and Lambada (LA). We also show the average accuracy across tasks (Avg). We see that for the Phi-3 model, LRC (69.7%) recovers a substantial portion of the FP16 accuracy (72%) relative to QuaRot (64.8%). The simpler SVD approach does *not* close the accuracy gap. For the larger models Llama-3 (8B)and Mixtral, we also see improvements of several percentage points.

To improve the accuracy of quantization, multiple approaches have considered use of groupsizing, where weight and activation matrices are divided into groups of size e.g. 128, and each group is scaled separately. This adds to the overall bitwidth, but improves accuracy. LRC can also be applied with groupsizing: we repeated the experiment in Table 1 in Table 2, this time applying a groups size of 128 to each method. Again, we see that LRC achieves multiple percentage-point improvements relative to QuaRot that are not possible with simple SVD.

4.2 ABLATION STUDIES

We perform three ablations to study the performances of LRC in different settings. We investigate the case where no quantization is performed on activations, and study the effect of the hyperparameter choices in LRC. In particular, we focus on the effect of the rank k and the choice of the quantizer used in line 5 of Algorithm 1.

Weights only. To examine the effects of loss due to activation quantization, we re-ran the experements presented above without quantizing the activations. We used the same set up as above, but we do no apply any quantizer operator in the forward pass (i.e. Q_a is set to be the identity map), such that no activation quantization is performed. Table 5 shows the performances. We see that all methods are able to recover (almost) perfectly the accuracy of the original models in the W4 regime. This experiment indicates that when only the weights are quantized, there is minimal error to correct for SoTA approaches, and as a result, low-rank terms do not provide any additional improvement.

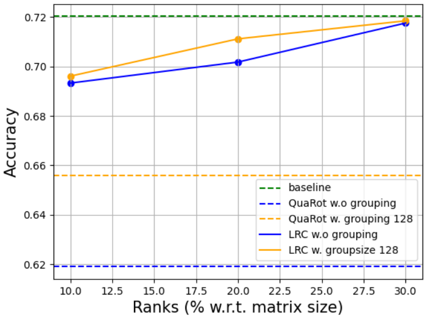

Figure 2: We show the effect of the rank, chosen as a percentage of the original weight matrices, on the performances of Phi-3 for lm-eval tasks when quantized at W4A4. We also show the effect of groupsizing activations. As baselines (dashed lines), we plot the performances of QuaRot with and without groupsizing, as well as the performance of the original model.

On the effect of Rank. In Figure 2, we show how the choice of the rank affects our proposed algorithm LRC 4 at W4A4. We compare the average accuracy obtained across all the tasks previously described when varying the rank, chosen as a percentage of the size of weight matrices. Note that this choice of rank is adaptive to the weight matrix, and ensures that the total overhead in memory is at most this percentage. We fix the LLM to be Phi-3 and we compare the performances obtained with the QuaRot baseline where no low-rank additional matrices are added. First we recover that even with ranks equal to 10% of the original matrix size, LRC outperforms QuaRot: we also see that if we allow ranks equal to 30% of the weight-size, LRC enables close-to-lossless performance. These results hold irrespective of the use of groupsizing.

On the effect of Quantizer. In Figure 3 we show how the effect of applying LRC using different quantizer in the update of $\widehat{\boldsymbol{W}}$ at W4A4 on Phi-3 . In Algorithm 1, we only require to have access to a solver of the layer-wise quantization problem. By default, we use GPTQ Frantar et al. (2022) in our main experiments, but here we aim at investigating the effect of additional low-rank matrices when using other quantizers, such as simple round-to-neatest (RTN) strategies to quantize weights. We observe that, LRC is always able to improve on its baseline version where no additional low-rank matrices are added, and this gap is even more pronounced when using simpler quantization strategies like RTN.

5 CONCLUSION & LIMITATIONS

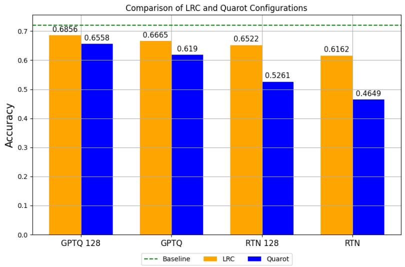

Figure 3: We show the effect of applying LRC with two quantization schemes, namely GPTQ and RTN, on the performances of Phi-3 on lm-eval tasks at W4A4.

We have studied low-rank corrections for LLM quantization. Our main innovation is to connect the low-rank matrix to the original, pre-quantized activations, whilst processing the quantized activations with a quantized weight matrix. Our method, LRC, solves jointly for the quantization and low rank structure. We have shown that a straight-forward approach to constructing the ranks, using SVD is not effective. Our method has the added complexity of computing activation statistics $\boldsymbol{\Sigma}$, but this allows us to significantly close the accuracy gap at W4A4 by incorporating low-rank weight matrices with ranks set to 10% of the original matrix sizes. To close the gap completely, we showed that on Phi-3 we needed ranks equal to 30% of the model size. Finally, we showed that LRC is composable with other quantization techniques, including groupsizing.

Limitations. In this work we have not studied the computational costs of adding low-rank computations to the forward pass. Some works have speculated that that the low-rank computation may

Method	Model	PPL	PQ	HS	A-e	A-c	WG	LA	Avg.
`FP16`	Phi-3	6.01	0.808	0.775	0.786	0.566	0.733	0.653	0.72
`QuaRot`		7.81	0.77	0.695	0.74	0.479	0.635	0.568	0.648
`SVD`		7.72	0.751	0.701	0.734	0.501	0.622	0.573	0.647
`LRC (1)`		7.26	**0.786**	0.731	0.796	**0.545**	**0.68**	**0.642**	**0.697**
`LRC (5)`		**7.2**	0.77	**0.734**	**0.799**	**0.545**	0.668	0.639	0.693
`FP16`	Llama-3 (8B)	6.13	0.807	0.792	0.778	0.533	0.726	0.76	0.733
`QuaRot`		7.78	0.765	0.74	0.721	0.441	0.663	0.704	0.672
`SVD`		**7.73**	0.769	**0.746**	0.697	0.46	0.68	0.699	0.675
`LRC (1)`		8.05	**0.773**	0.736	0.749	0.476	**0.707**	0.731	0.695
`LRC (5)`		7.94	0.764	0.742	**0.758**	**0.483**	0.705	**0.739**	**0.698**
`FP16`	Mixtral	3.84	0.837	0.84	0.834	0.596	0.766	0.784	0.776
`QuaRot`		4.55	0.813	**0.814**	0.794	**0.569**	0.726	0.746	0.744
`SVD`		4.51	**0.817**	**0.814**	0.802	0.559	0.726	0.761	0.746
`LRC (1)`		4.42	0.81	0.801	0.811	0.561	0.724	**0.818**	**0.754**
`LRC (5)`		**4.41**	0.801	0.8	**0.813**	0.555	**0.736**	0.814	0.753

Table 1: Accuracy on LLM-EVAL with weight and activation quantization (W4A4) and no group-scaling. LRC and SVD methods use low-rank matrices with 10% of the orignal matrix ranks. We have highlighted in bold the best performances among the quantized models.

Method	Model	PPL	PQ	HS	A-e	A-c	WG	LA	Avg.
`FP16`	Phi-3	6.01	0.808	0.775	0.786	0.566	0.733	0.653	0.72
`QuaRot`		7.65	0.778	0.7	0.768	0.511	0.665	0.548	0.661
`SVD`		7.54	0.77	0.696	0.751	0.52	0.666	0.555	0.659
`LRC (1)`		7.28	**0.786**	0.722	**0.815**	**0.567**	0.693	0.644	**0.704**
`LRC (5)`		**7.25**	0.776	**0.728**	0.797	0.539	**0.706**	**0.65**	0.699
`FP16`	Llama-3 (8B)	6.13	0.807	0.792	0.778	0.533	0.726	0.76	0.733
`QuaRot`		7.42	0.782	0.747	0.75	0.469	0.712	0.731	0.699
`SVD`		7.36	0.779	0.759	0.762	0.479	0.72	0.717	0.703
`LRC (1)`		7.03	0.78	**0.762**	**0.77**	**0.505**	0.715	0.764	0.716
`LRC (5)`		**7.02**	**0.783**	0.761	0.766	0.494	**0.735**	**0.765**	**0.717**
`FP16`	Mixtral	3.84	0.837	0.84	0.834	0.596	0.766	0.784	0.776
`QuaRot`		4.44	**0.822**	0.816	0.809	**0.578**	0.736	0.763	0.754
`SVD`		4.41	0.821	**0.821**	0.818	0.574	**0.747**	0.765	0.758
`LRC (1)`		4.26	0.816	0.811	0.815	0.567	0.729	**0.821**	0.76
`LRC (5)`		**4.25**	0.817	0.812	**0.817**	0.572	0.738	0.815	**0.762**

Table 2: Accuracy on LLM-EVAL with weight and activation quantization (W4A4). For each method we use a groupsize of 128 for both weights and activations. We have highlighted in bold the best performances among the quantized models.

be computable in parallel with the low-bitwidth computation: we leave such an implementation and a thorough study to future work.

We found that running our LRC procedure for multiple iterations did not comprehensively improve the performance. We found that a single iteration was often sufficient, and anecdotally we found that convergence was dependent on the damping factors used in Cholesky computations. We speculate that larger calibration set may improve the condition of the Hessians.

Finally, our work highlights that for W4A4, there is significant information lost in quantizing activations. We have followed previous works in using a scale-then-round scheme, with hyper-parameter search for the best scale. The need to perform activation quantization on-the-fly means that fast (simple!) schemes are needed. this appears to be a productive direction for future improvements.

Method	Model	PPL	PQ	HS	A-e	A-c	WG	LA	Avg.
`FP16`	Phi-3	6.01	0.808	0.775	0.786	0.566	0.733	0.653	0.72
`QuaRot`		6.3	0.804	0.756	**0.781**	0.561	0.719	0.642	0.711
`SVD`		6.24	**0.808**	0.759	0.779	**0.567**	**0.727**	**0.646**	**0.714**
`LRC`		**6.21**	0.805	**0.76**	0.772	0.558	0.723	0.641	0.71
`FP16`	Llama-3 (8B)	6.13	0.807	0.792	0.778	0.533	0.726	0.76	0.733
`QuaRot`		6.55	**0.805**	0.779	**0.774**	**0.519**	**0.742**	0.74	**0.727**
`SVD`		6.49	0.799	**0.783**	0.765	0.508	0.738	**0.749**	0.724
`LRC`		**6.47**	0.8	0.78	0.761	0.51	0.731	0.747	0.722
`FP16`	Mixtral	3.84	0.837	0.84	0.834	0.596	0.766	0.784	0.776
`QuaRot`		3.98	0.836	0.836	0.825	**0.593**	0.757	0.781	0.771
`SVD`		3.96	0.835	0.837	0.832	**0.593**	**0.766**	**0.787**	**0.775**
`LRC`		**3.95**	**0.84**	**0.839**	**0.825**	**0.593**	0.754	0.783	0.772

Table 3: Accuracy on LLM-EVAL with weight only quantization. We have highlighted in bold the best performances among the quantized models.

REFERENCES

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. Towards end-to-end 4-bit inference on generative large language models. *arXiv preprint arXiv:2310.09259*, 2023.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *arXiv preprint arXiv:2404.00456*, 2024.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35: 30318–30332, 2022.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. Spqr: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization. *arXiv preprint arXiv:2401.06118*, 2024.

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.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024. URL `https://zenodo.org/records/12608602`.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2704–2713, 2018.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. Gear: An efficient kv cache compression recipefor near-lossless generative inference of llm. *arXiv preprint arXiv:2403.05527*, 2024.

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.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant–llm quantization with learned rotations. *arXiv preprint arXiv:2405.16406*, 2024.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pp. 7197–7206. PMLR, 2020.

Lin Ou, Jinpeng Xia, Yuewei Zhang, Chuzhan Hao, and Hao Henry Wang. Adaptive quantization error reconstruction for llms with mixed precision. In *First Conference on Language Modeling*, 2024.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*, 2023.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks. *arXiv preprint arXiv:2402.04396*, 2024.

Mart van Baalen, Andrey Kuzmin, Markus Nagel, Peter Couperus, Cedric Bastoul, Eric Mahurin, Tijmen Blankevoort, and Paul Whatmough. Gptvq: The blessing of dimensionality for llm quantization. *arXiv preprint arXiv:2402.15319*, 2024.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. *Advances in Neural Information Processing Systems*, 35:17402–17414, 2022.

Xiaoxia Wu, Cheng Li, Reza Yazdani Aminabadi, Zhewei Yao, and Yuxiong He. Understanding int4 quantization for transformer models: Latency speedup, composability, and failure cases. *arXiv preprint arXiv:2301.12017*, 2023.

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.

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

Cheng Zhang, Jianyi Cheng, George A Constantinides, and Yiren Zhao. Lqer: Low-rank quantization error reconstruction for llms. *arXiv preprint arXiv:2402.02446*, 2024.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate llm serving. *Proceedings of Machine Learning and Systems*, 6:196–209, 2024.

APPENDIX

.1 PROOF OF PROPOSITION 3.1

By simply developing the objective and discarding the constant term (i.e. those independent of $\widehat{\boldsymbol{W}}$), we can reformulate the objective function as:

$$\begin{aligned}&\langle \widehat{\boldsymbol{W}}, \widehat{\boldsymbol{W}}\boldsymbol{\Sigma}_y\rangle - 2[\langle \widehat{\boldsymbol{W}}, (\boldsymbol{W} - \boldsymbol{U}^{(t)}(\boldsymbol{V}^{(t)})^\top)\boldsymbol{X}\boldsymbol{Y}^\top\rangle]\\ &= \langle \widehat{\boldsymbol{W}}, \widehat{\boldsymbol{W}}\boldsymbol{\Sigma}_y\rangle - 2[\langle \widehat{\boldsymbol{W}}, (\boldsymbol{W} - \boldsymbol{U}^{(t)}(\boldsymbol{V}^{(t)})^\top)\boldsymbol{X}\boldsymbol{Y}^\top\boldsymbol{\Sigma}_y\boldsymbol{\Sigma}_y^{-1}\rangle]\end{aligned}$$

where $\boldsymbol{\Sigma}_y := \boldsymbol{Y}\boldsymbol{Y}^\top$, and the second equality holds under the the assumption that $\boldsymbol{Y}$ is full rank with $n \geq d_{\text{in}}$. Then by denoting

$$\widetilde{\boldsymbol{W}}^{(t)} := (\boldsymbol{W} - \boldsymbol{U}^{(t)}(\boldsymbol{V}^{(t)})^\top)\boldsymbol{X}\boldsymbol{Y}^\top(\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1}$$

we obtain that the objective is equivalent to

$$\langle \widehat{\boldsymbol{W}} - \widetilde{\boldsymbol{W}}^{(t)}, (\widehat{\boldsymbol{W}} - \widetilde{\boldsymbol{W}}^{(t)})\boldsymbol{\Sigma}_y\rangle$$

which conclude the proof.

.2 PROOF OF PROPOSITION 3.4

Let us write the first order condition for $\boldsymbol{V}$. Indeed we obtain that:

$$\boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{V}^\top\boldsymbol{\Sigma}_x = \boldsymbol{U}^\top[\boldsymbol{W}\boldsymbol{\Sigma}_x - \widehat{\boldsymbol{W}}\boldsymbol{Y}\boldsymbol{X}^\top]$$

where $\boldsymbol{\Sigma}_x := \boldsymbol{X}\boldsymbol{X}^\top$ Under the assumption that $\boldsymbol{X}$ is full rank with $n \geq d_{\text{in}}$, we deduce that

$$\boldsymbol{U}^\top\boldsymbol{U}\boldsymbol{V}^\top = \boldsymbol{U}^\top[\boldsymbol{W} - \widehat{\boldsymbol{W}}\boldsymbol{Y}\boldsymbol{X}^\top\boldsymbol{\Sigma}_x^{-1}]$$

which always admits a solution. Then by denoting $(\boldsymbol{U}^\top\boldsymbol{U})^{-1}$ the Moore–Penrose inverse of $\boldsymbol{U}^\top\boldsymbol{U}$, we obtain that:

$$\boldsymbol{V}^\top = (\boldsymbol{U}^\top\boldsymbol{U})^{-1}\boldsymbol{U}^\top[\boldsymbol{W} - \widehat{\boldsymbol{W}}\boldsymbol{Y}\boldsymbol{X}^\top\boldsymbol{\Sigma}_x^{-1}]$$

Then by plugging this expression into the original objective, we obtain the following equivalent optimization problem:

$$\min_{\boldsymbol{U}\in\mathbb{R}^{d^{\text{out}}\times k}} \|\boldsymbol{W}\boldsymbol{X} - \widehat{\boldsymbol{W}}\boldsymbol{Y} - \boldsymbol{U}(\boldsymbol{U}^\top\boldsymbol{U})^{-1}\boldsymbol{U}^\top[\boldsymbol{W} - \widehat{\boldsymbol{W}}\boldsymbol{Y}\boldsymbol{X}^\top\boldsymbol{\Sigma}_x^{-1}]\boldsymbol{X}\|_F^2$$

and as $\{\boldsymbol{U}(\boldsymbol{U}^\top\boldsymbol{U})^{-1}\boldsymbol{U}^\top :\ \ \boldsymbol{U} \in \mathbb{R}^{d^{\text{out}}\times k}\}$ spans the space of orthogonal projection onto subspaces of dimension at most k, we can reparameterize the optimization problem as:

$$\min_{\boldsymbol{U}\in\mathbb{R}^{d^{\text{out}}\times k}\cap\mathcal{O}} \|\boldsymbol{W}\boldsymbol{X} - \widehat{\boldsymbol{W}}\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{U}^\top[\boldsymbol{W} - \widehat{\boldsymbol{W}}\boldsymbol{Y}\boldsymbol{X}^\top\boldsymbol{\Sigma}_x^{-1}]\boldsymbol{X}\|_F^2$$

Then by developing again this expression and discarding the constant terms (i.e. those independent of $\boldsymbol{U}$), we recover exactly the optimization problem defined in Proposition 3.4. Finally observe $\boldsymbol{\Sigma} := \boldsymbol{\Sigma}_{\boldsymbol{1}} + \boldsymbol{\Sigma}_{\boldsymbol{2}}^{(t)} - \boldsymbol{\Sigma}_{\boldsymbol{3}}^{(t)}$ is symmetric but not necessarily positive semi-definite. However, observe that for any symmetric matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$, $\boldsymbol{U} \in \mathbb{R}^{d\times k}$ with orthonormal columns and $\alpha > 0$, we have:

$$\text{Tr}(\boldsymbol{U}^\top\boldsymbol{\Sigma}\boldsymbol{U}) = \text{Tr}(\boldsymbol{U}^\top(\boldsymbol{\Sigma} + \alpha\text{I}_{\text{d}})\boldsymbol{U}) - \alpha\text{Tr}(\boldsymbol{U}^\top\boldsymbol{U}) = \text{Tr}(\boldsymbol{U}^\top(\boldsymbol{\Sigma} + \alpha\text{I}_d)\boldsymbol{U}) + k\alpha$$

from which we deduces the that $\boldsymbol{U}$ can be chosen to be the k first unit eigenvectors of $\boldsymbol{\Sigma}$, which concludes the proof.

.3 PROOF OF PROPOSITION 3.5

Observe that we can rewrite the optimization problem as:

$$\min_{\boldsymbol{U}\in\mathbb{R}^{d^{\text{out}}\times k}, \boldsymbol{V}\in\mathbb{R}^{d^{\text{in}}\times k}} \ \min_{\widetilde{\boldsymbol{W}}\in\mathbb{R}^{d^{\text{out}}\times d^{\text{in}}}} \mathcal{L}_{\text{qlr}}(\widetilde{\boldsymbol{W}}, \boldsymbol{U}, \boldsymbol{V})\ .$$

Now given $\boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}, \boldsymbol{V} \in \mathbb{R}^{d^{\text{in}} \times k}$, we can derive the first order condition of the inner optimization problem, that is:

$$\frac{\partial \mathcal{L}_{\text{qlr}}}{\partial \widetilde{\boldsymbol{W}}} = 0 \quad \text{which gives} \quad \widetilde{\boldsymbol{W}} \boldsymbol{\Sigma}_y = [\boldsymbol{W} - \boldsymbol{U}\boldsymbol{V}^\top] \boldsymbol{X}\boldsymbol{Y}^\top$$

where $\boldsymbol{\Sigma}_y := \boldsymbol{Y}\boldsymbol{Y}^\top$. Now under the assumption that $\boldsymbol{Y} \in \mathbb{R}^{d_{\text{in}} \times n}$ is full rank where $n \geq d_{\text{in}}$, we obtain that

$$\widetilde{\boldsymbol{W}} = [\boldsymbol{W} - \boldsymbol{U}\boldsymbol{V}^T] \boldsymbol{X}\boldsymbol{Y}^T \boldsymbol{\Sigma}_y^{-1}$$

from which we deduce the equivalent optimization problem:

$$\min_{\boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}, \boldsymbol{V} \in \mathbb{R}^{d^{\text{in}} \times k}} \|(\boldsymbol{W} - \boldsymbol{U}\boldsymbol{V}^\top) \tilde{\boldsymbol{X}}\|_F^2$$

where $\tilde{\boldsymbol{X}} := \boldsymbol{X} - \boldsymbol{X}\boldsymbol{Y}^\top \boldsymbol{\Sigma}_y^{-1} \boldsymbol{Y}$. Again, by fixing $\boldsymbol{U}$ and by deriving the first order condition for $\boldsymbol{V}$, we obtain that:

$$\boldsymbol{U}^\top \boldsymbol{U} \boldsymbol{V}^\top \tilde{\boldsymbol{X}} \tilde{\boldsymbol{X}}^\top = \boldsymbol{U}^\top \boldsymbol{W} \tilde{\boldsymbol{X}} \tilde{\boldsymbol{X}}^\top$$

Assuming that $\tilde{\boldsymbol{X}}$ is full rank, we recover the normal equation

$$\boldsymbol{U}^\top \boldsymbol{U} \boldsymbol{V}^\top = \boldsymbol{U}^\top \boldsymbol{W}$$

which always admit a solution. Then by denoting $(\boldsymbol{U}^\top \boldsymbol{U})^{-1}$ the Moore–Penrose inverse of $\boldsymbol{U}^\top \boldsymbol{U}$, we obtain that:

$$\boldsymbol{V}^\top = (\boldsymbol{U}^\top \boldsymbol{U})^{-1} \boldsymbol{U}^\top \boldsymbol{W}$$

Plugging back this expression to the previous objective leads to the following optimization problem:

$$\min_{\boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}} \|(\boldsymbol{I}_d - \boldsymbol{U}(\boldsymbol{U}^\top \boldsymbol{U})^{-1} \boldsymbol{U}^\top) \boldsymbol{W} \tilde{\boldsymbol{X}}\|_F^2$$

and by denoting $\boldsymbol{O} := \boldsymbol{W}\tilde{\boldsymbol{X}}$, we recover the PCA of $\boldsymbol{O}$, that is:

$$\min_{\boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}} \|(\boldsymbol{I}_d - \boldsymbol{U}(\boldsymbol{U}^\top \boldsymbol{U})^{-1} \boldsymbol{U}^\top) \boldsymbol{O}\|_F^2$$

as $\{\boldsymbol{U}(\boldsymbol{U}^\top \boldsymbol{U})^{-1} \boldsymbol{U}^\top : \ \boldsymbol{U} \in \mathbb{R}^{d^{\text{out}} \times k}\}$ spans the space of orthogonal projection onto subspaces of dimension at most k. Now observe that

$$\boldsymbol{O}\boldsymbol{O}^\top = \boldsymbol{W}\boldsymbol{X}[I_n - \boldsymbol{Y}^\top (\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1} \boldsymbol{Y}] \boldsymbol{X}^\top \boldsymbol{W}^\top = \boldsymbol{\Sigma}_{\text{init}}$$

which conclude the proof.