Parameter and Memory Efficient Language Model Compression using Fisher Information from Low-Rank Representations

Anonymous ACL submission

Abstract

Modern language models demonstrate excellent performance in diverse text processing tasks. Yet, to achieve the best quality, memory and computationally demanding fine-tuning on a downstream task is required. While PEFT methods, such as LoRA enable almost no VRAM overhead for fine-tuning, the amount of memory and compute may be still prohibitive for the regular users. To compress and speed up LMs pruning techniques, such as Fisher-Weighted Singular Value Decomposition (FWSVD) (Hsu et al., 2022) are therefore additionally used. Yet, FWSVD requires a downstream task fine-tuning to gather Fisher information. Our work tries to break this vicious circle of dependence on large expensive GPU showing that state-of-the-art LM compression, such as FWSVD, can be done without storing the full gradients. Namely, our approach combines the reduced number of training parameters up to $0,01\%$ of the initial amount of parameters and the VRAM utilization up to 15%, for a pruning 20% of the fine-tuned model weights without any noticeable loss of accuracy. We evaluate this approach on various tasks including NLU, NER, MMLU, and summarization demonstrating the effectiveness of the method as compared to strong baselines.

1 Introduction

The recent emergence of transformers and Large language models (LLM) has revolutionized the field of natural language processing. These models, with their ability to solve a wide range of tasks and generate human-like language, have demonstrated outstanding performance in a variety of NLP tasks. However, the growing size of these models poses significant challenges. They require enormous amounts of computational resources and energy to train and run, making it difficult to fit them on most consumer GPUs.

Therefore, there is an urgent need to reduce the size of language models or make their fine-tuning computationally affordable. These methods require several cycles of training the full-size model to gather the necessary information. In the case of large models, these cycles can be resource-intensive, as the training loop requires doubling the memory occupied: we store both the parameters and their gradients.

We propose a solution to effectively reduce model size without the need to perform full-scale fine-tuning. We combine the Fisher-Weighted Singular Value Decomposition (Hsu et al., 2022) and parameter-efficient fine-tuning LoRA approaches by training only low-rank representations of the fully connected part of the model and collecting gradients for FWSVD only for these layers. We achieve almost similar compression quality as in the original FWSVD approach but with 0.001% of the initial amount of parameters. Summarizing, the contributions of this work are as follows:

1. We propose a *novel approach* for a parameter and memory-efficient pruning based on Fisher Information collected from Low-Rank representations.

2. We test our approach in two different setups: *with* and *without* GPU memory limits, showing that our compression can be done even without direct fine-tuning on a downstream task.

3. We perform a comprehensive experimental study of the proposed approach comparing it to the state-of-the-art baselines observing *no substantial performance drop* across a wide range of NLP tasks and language models.

2 Related Work

Great performance of NLP models comes along with their enormous size. Most of the current state-of-the-art LLMs have at least 7 billion parameters (Touvron et al., 2023; Jiang et al., 2023; Chiang

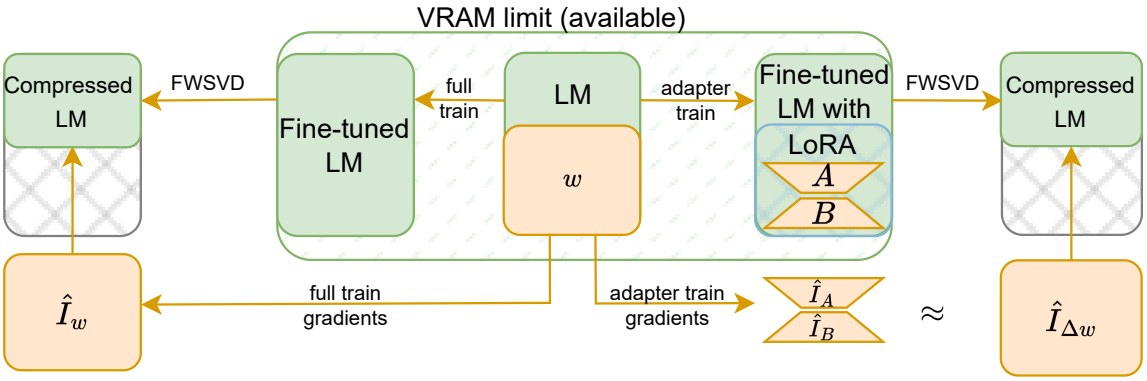

Figure 1: Workflows of the original FWSVD-based compression and the proposed approach FWSVD+LoRA which saves peak GPU RAM during the compression by approximating full model gradients using LoRA.

et al., 2023) making it nearly impossible to fine-tune them even on high-end consumer GPUs. To avoid the challenges posed by the enormous size of the current state-of-the-art LLMs, researchers have explored various techniques for model compression and Parameter-Efficient Fine-Tuning (PEFT).

Various methods, such as quantization (Dettmers et al., 2022; Zhang et al., 2023) or pruning (Hu et al., 2021; Hsu et al., 2022; Xia et al., 2023; Kurtic et al., 2023) are being proposed by the researchers to allow the inference and fine-tuning of modern LLMs in a limited computational budge.

To achieve efficient compression while preserving quality, techniques that assess parameter impact on task response are usually employed. Examples include Fisher information (Hsu et al., 2022; Hua et al., 2022), Layer-Wise Relevance Propagation (Fan et al., 2021; Voita et al., 2019), or Saliency Score (Kurtic et al., 2023; Ma et al., 2023).

Also, a variety of parameter-efficient fine-tuning (PEFT) (Hu et al., 2022; Zaken et al., 2022; Liu et al., 2022), that allow to train only several thousand out of billions of model parameters and yet provide good performance were proposed recently.

In this context, our proposed approach combines the FWSVD and parameter-efficient fine-tuning LoRA methods. By training only low-rank representations of the fully connected part of the model and collecting gradients for FWSVD only for these layers, we achieve significant compression without sacrificing performance.

3 Background

In this section, we give a comprehensive introduction to our proposed method based on low-rank approximations from previous works.

Model	Num. of Params	VRAM (GiB)
`roberta-base`: full	355 (100%)	3.80 (100%)
$\texttt{LoRA}_{r=1}$	1.9 (0.0054%)	2.92 (77%)
$\texttt{LoRA}_{r=8}$	3.7 (0.0105%)	2.93 (77%)
$\texttt{QLoRA}_{r=1}$	1.9 (0.0054%)	1.14 (30%)
$\texttt{QLoRA}_{r=8}$	3.7 (0.0105%)	1.15 (30%)
`t5-large`: full	770 (100%)	22.00 (100%)
$\texttt{LoRA}_{r=8}$	4.1 (0.0053%)	16.84 (77%)
$\texttt{LoRA}_{r=32}$	16.0 (0.0208%)	16.95 (77%)
$\texttt{QLoRA}_{r=8}$	4.1 (0.0053%)	3.12 (14%)
$\texttt{QLoRA}_{r=32}$	16.0 (0.0208%)	3.22 (15%)
`mistral 7b`: full	7270 (100%)	64.82 (100%)
$\texttt{LoRA}_{r=16}$	28.31 (0.0039%)	53.17 (82%)
$\texttt{QLoRA}_{r=16}$	28.31 (0.00389%)	9.82 (15%)
$\texttt{QLoRA}_{r=64}$	11.32 (0.0156%)	9.88 (15%)

Table 1: Fraction of trainable parameters (in millions and %) and GPU utilization) for fine-tuning of the whole model and with LoRA/QLoRA.

3.1 SVD: Singular Value Decomposition

Singular Value Decomposition (SVD) is one of the simplest low-rank decomposition methods and has been widely used to prune word embeddings (Lan et al., 2020) and transformer models (Michel et al., 2019; Hu et al., 2021). Considering w as the weight matrix of the model, the SVD for this matrix will be $w = U\Sigma V^{\mathrm{T}}$. We then perform truncation by rank r: $U_r = U[:, :r]$, $\Sigma_r = \Sigma[:r, :r]$, $V_r = V[:, :r]$, and split w into two low-rank matrices $w_1 = \sqrt{\Sigma_r} V_r^{\mathrm{T}}$ and $w_2 = U_r \sqrt{\Sigma_r}$. Hence, $w \approx w_2 w_1$. If w was of shape (m, n) and had $m \times n$ parameters, then after SVD, it becomes $r \times (m + n)$. As the number of parameters goes down, forward pass time for the models reduces as well. We demonstrate it on Figure 2 for `t5-large` model. In addition, SVD compression does not require special algorithms or libraries (like pruning) and can be used together with other methods for

evaluation speed-up. However, SVD-based compression typically results in poor performance at various ranks, except for those closest to the original matrix rank.

3.2 Fisher Information

Fisher information denoted I_w^{def}, is a measure that quantifies the amount of information a data set $\mathcal{D}$ provides about a weight matrix w in a model:

$$I_w^{\text{def}} = \mathbb{E}\left[\left(\frac{\partial}{\partial w}\log p(\mathcal{D}|w)\right)^2\right] \quad (1)$$

Its computation is challenging due to the need for taking the expectation over the entire data distribution (Murphy, 2012). Therefore, various approximation methods have been proposed in the literature, such as the use of score function estimators (Casella, 2001), or the use of natural gradient descent which implicitly approximates the Fisher information matrix (Amari, 1998). The latter while not giving any theoretical guarantees (Kunstner et al., 2019), works well in practice.

3.3 FWSVD: Fisher-Weighted SVD

Despite all advantages, SVD-based model compression typically results in poor performance at various ranks, except for those closest to the original matrix rank. Therefore, Hsu et al. (2022) proposed an improved SVD-based compression approach using empirical Fisher information, which excels in compression performance. To address this limitation, Hsu et al. (2022) introduces an extension to the conventional SVD approach, which integrates empirical Fisher information into the SVD decomposition. This method is named Fisher-Weighted SVD (FWSVD) and significantly outperforms standard SVD compression.

As described in Section 3.2, the direct estimation of Fisher information is intractable. Therefore, following previous work (Amari, 1998; Pascanu and Bengio, 2014; Hsu et al., 2022), we estimate the empirical Fisher information in this work. Let I_w^{emp} be the empirical Fisher information matrix w.r.t. to weight matrix w, we calculate:

$$I_w^{\text{def}} \approx I_w^{\text{emp}} = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\left(\frac{\partial}{\partial w}\mathcal{L}\left(d_i; w\right)\right)^2 \quad (2)$$

Given that $\hat{w} = \text{SVD}(w)$, SVD-based low-rank approximation problem transforms into:

$$\min_{\text{rank}\,\hat{w}=r} \|\sqrt{I_w^{\text{emp}}} * (w - \hat{w})\|^2 \quad (3)$$

Problem 3 does not have a closed-form solution (Srebro and Jaakkola, 2003). To avoid this issue, Hsu et al. (2022) suggest solving the low-rank approximation problem using row-wise weighting:

$$\hat{I}_w^{\text{emp}} = \text{diag}\left(I_w^{\text{emp}} \cdot \mathbf{1}\right) \quad (4)$$

As a result, we get weighted SVD components:

$$\text{FWSVD}(w) \approx \hat{U}\hat{\Sigma}\hat{V} = (\hat{I}_w^{\text{emp}})^{-1}U\Sigma V \quad (5)$$

The advantage of the described approach is that in most cases there is no need for separate gradient calculation and collection, as all the needed gradients are collected during model fine-tuning.

3.4 LoRA: Low Rank Adaptation

Since most current language models have a reasonable general understanding of language, they often require additional fine-tuning on a downstream task to achieve the desired performance. This is where the Low-Rank Adaptation (LoRA) approach Hu et al. (2022). Hu et al. (2022) introduces a very computationally efficient way to train language models using only a few percent of the model's parameters. Moreover, the trained parameters are then fed back into the model without the need to store additional weights for inference. With the advent of large language models such as LLaMa (Touvron et al., 2023), Mistral (Jiang et al., 2023), and others, LoRA has become a standard technique for tuning LLMs for a downstream task. The most common pipeline is as follows: the LLM itself is quantized to half-precision (fp16) or even more (int-4), and the LoRA module is trained on a given downstream task.

The authors of LoRA restrict the fine-tuning of a weight matrix $w \in \mathbb{R}^{n\times m}$ to the tuning of its low-rank decomposition $w + \Delta w = w + BA$, where $A \in \mathbb{R}^{n\times r}, B \in \mathbb{R}^{r\times m}$, and $r \in \mathbb{R} : r \ll \min(n, m)$. Only Δw is updated during the fine-tuning. The forward pass for input x is simple: $h = wx + \Delta wx = wx + BAx$.

4 Our Approach: LoRA-enhanced FWSVD Compression

We propose a novel approach that synergizes the advantages of LoRA (low-rank adaptation) and FWSVD while mitigating their respective drawbacks. The key idea is that empirical Fisher information matrices can be effectively obtained based solely on LoRA representations, enabling FWSVD

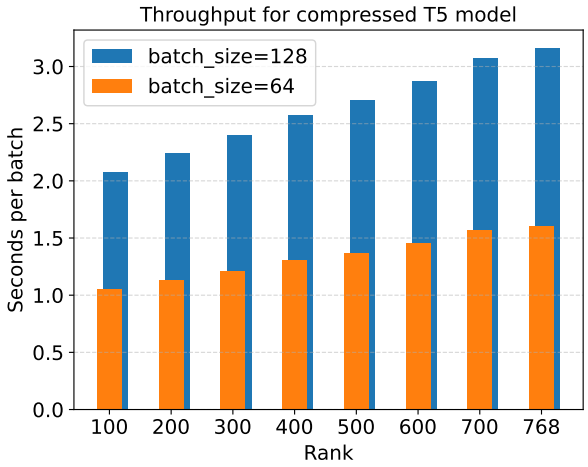

Figure 2: Speed-up of batch processing for the compressed `t5-large` model. The last bar is the speed for the uncompressed model.

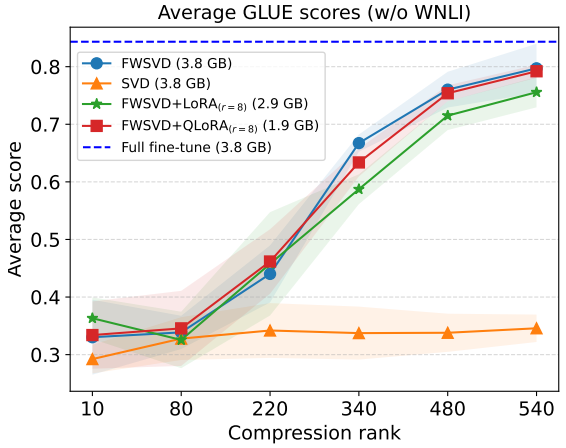

Figure 3: Results on the GLUE benchmark for different compression ranks of `roberta-base` model. Numerical results are in Tables 4 and 6.

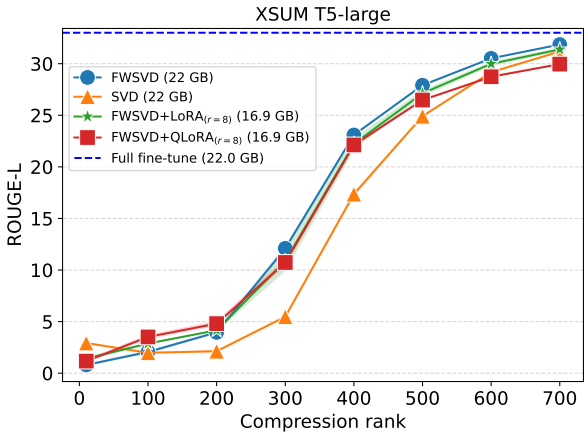

Figure 4: Results on the XSUM summarization dataset for *VRAM-limited* compression setup of `t5-large` model. Numerical results are in Tables 2 and 9.

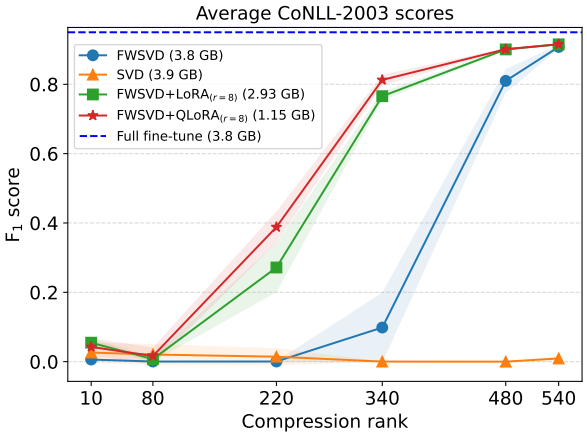

Figure 5: Average F_1 scores on the CoNLL-2003 dataset for different compression ranks of `roberta-base`. Additional results are in Tables 4 and 8.

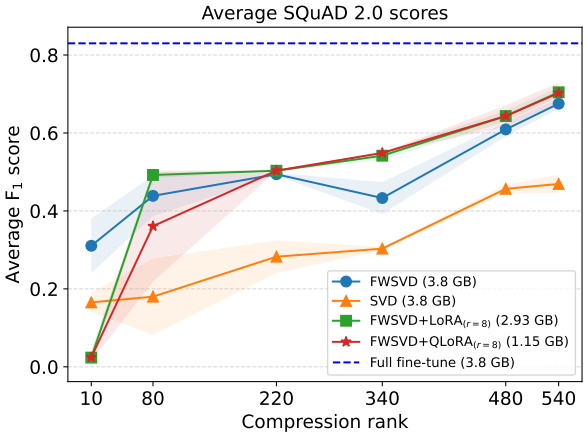

Figure 6: Average F_1 scores on the SQUAD v2 dataset for different compression ranks of `roberta-base`. Additional results are in Tables 4 and 8.

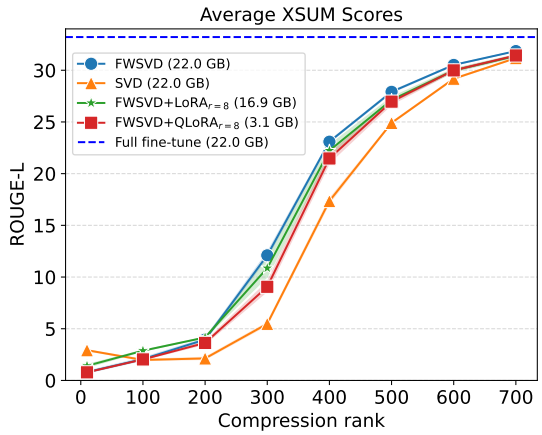

Figure 7: Results on the XSUM summarization dataset for *VRAM-unlimited* compression setup of `t5-large` model. Numerical results are in Tables 3 and 11.

to perform comparably to its original implementation. This approach eliminates the need for extensive gradient computation and storage, resulting in a more efficient and scalable pruning process. Indeed, given a large LLM its compression with FWSVD will require storing its gradients, which often doubles the amount of required GPU memory, while our method do not require this extra memory for to perform the compression. The proposed approach achieves a performance level similar to standard FWSVD while obviating its memory-intensive requirements and the need for multiple rounds of fine-tuning. This advancement paves the way for more practical and efficient model pruning in natural language processing applications.

So, instead of calculating $\hat{I}_w^{\text{emp}}$ for each weight matrix w we will compress, we will calculate

$$\hat{I}_w^{\text{emp}} \approx \hat{I}_{\Delta w}^{\text{emp}} = \hat{I}_B^{\text{emp}} \hat{I}_A^{\text{emp}} \tag{6}$$

After we fine-tune the model with LoRA, we merge Δw back into the original weight matrix w and compress it using $\hat{I}_{\Delta w}$:

$$\text{FWSVD}(w) \approx \text{FWSVD}(\Delta w) \tag{7}$$

$$= \text{SVD}(\hat{I}_{\Delta w}^{\text{emp}}(w + \Delta w)) \tag{8}$$

$$= \text{SVD}(\hat{I}_{\Delta w}^{\text{emp}} w) \tag{9}$$

In this way, we only need to train less than 0.01% of model parameters, yet we are still able to achieve compression performance comparable to the original FWSVD approach.

Original FWSVD additionally proposes an algorithm that performs best in the "fine-tune $\rightarrow$ compress $\rightarrow$ fine-tune" paradigm, requiring two rounds of fine-tuning. This algorithm will not be used in this paper as we leave such experiments for future work. In this work, we consider the "fine-tune $\rightarrow$ compress" approach, which, nevertheless, demonstrates satisfactory performance on most NLP tasks.

5 Experimental Setup

In this section, we show our experimental setup. We begin by introducing the datasets and benchmarks used for evaluating our proposed approach and baselines. Next, we detail the models employed in our evaluation, along with links to their corresponding checkpoints. We then discuss two compression pipelines: one with unlimited VRAM and another with limited VRAM. Additionally, we specify the layers compressed in each model.

Model size	72%	79%	85%	92%
Method/Rank	400	500	600	700
SVD	16.3$_{\pm 0.02}$	24.7$_{\pm 0.03}$	29.3$_{\pm 0.01}$	31.4$_{\pm 0.02}$
FWSVD	**22.9**$_{\pm 0.01}$	**27.8**$_{\pm 0.01}$	**30.4**$_{\pm 0.01}$	**31.8**$_{\pm 0.01}$
FWSVD+LoRA$_{r=8}$	22.4$_{\pm 0.01}$	27.2$_{\pm 0.01}$	30.0$_{\pm 0.01}$	31.4$_{\pm 0.01}$
FWSVD+QLoRA$_{r=8}$	21.5$_{\pm 0.01}$	27.0$_{\pm 0.01}$	30.0$_{\pm 0.01}$	31.4$_{\pm 0.01}$

Table 2: Average performance on XSUM for `t5-large` model in the VRAM-unlimited setup. Extended results are in Table 9. Row *model size* represents the fraction of the uncompressed model size left after compression. Full rank is 768.

Model size	72%	79%	85%	92%
Method/Rank	400	500	600	700
SVD	17.5$_{\pm 0.39}$	25.0$_{\pm 0.10}$	29.2$_{\pm 0.13}$	31.2$_{\pm 0.07}$
FWSVD	**23.6**$_{\pm 0.15}$	**28.1**$_{\pm 0.17}$	**30.5**$_{\pm 0.08}$	**31.9**$_{\pm 0.02}$
FWSVD+LoRA$_{r=8}$	22.2$_{\pm 0.24}$	26.5$_{\pm 0.09}$	28.7$_{\pm 0.08}$	29.9$_{\pm 0.06}$
FWSVD+QLoRA$_{r=8}$	22.4$_{\pm 0.61}$	26.3$_{\pm 0.40}$	28.5$_{\pm 0.14}$	29.84$_{\pm 0.13}$

Table 3: Average performance on XSUM for `t5-large` model in the *VRAM-limited* setup. Extended results are in Table 11. Row *model size* represents the fraction of the uncompressed model size left after compression. Full rank is 768.

5.1 Datasets

We evaluate the performance of our proposed method compared to standard SVD, FWSVD, and standard LoRA approaches. For the experiments, we use the RoBERTa model (Liu et al., 2019) and test its performance on GLUE benchmark (Wang et al., 2019) (we exclude the WNLI dataset from the evaluation and discuss the reasons in the Appendix A.2), CoNLL-2003 (Sang and Meulder, 2003) and SQuAD v**2.0** (Rajpurkar et al., 2016) datasets. We also conduct experiments on the summarization dataset XSUM (Narayan et al., 2018) with the T5 language model (Raffel et al., 2020) and evaluate the 7B version of Mistral v2 (Jiang et al., 2023) on the MMLU benchmark (Hendrycks et al., 2021) using 5-shot evaluation.

5.2 Models

In our classification experiments, we use `roberta-base` model[1] from the HuggingFace model hub. We set the batch size to 32 and 128 during training and evaluation. In each task, we fine-tune the model for 3 epochs using AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate equal to $3 \cdot 10^{-5}$. For LoRA we set rank r to 1 and 8, and scale s to 2 and 16

[1] huggingface.co/roberta-base

Benchmark	GLUE Average scores			CoNLL-2003 F_1 scores			SQuAD v2 F_1 scores		
Model Size	80%	90%	95%	80%	90%	95%	80%	90%	95%
Method/Rank	340	480	540	340	480	540	340	480	540
SVD	0.34±0.04	0.34±0.03	0.35±0.02	0.00±0.00	0.03±0.02	0.02±0.02	0.30±0.00	0.46±0.01	0.47±0.02
FWSVD	**0.67**±0.02	**0.76**±0.03	**0.80**±0.04	0.10±0.08	0.81±0.03	0.91±0.00	0.48±0.03	0.57±0.02	0.64±0.01
FWSVD+LoRA$_{r=8}$	0.59±0.03	0.71±0.03	0.76±0.03	**0.83**±0.01	**0.92**±0.00	**0.93**±0.00	0.53±0.01	0.59±0.01	0.64±0.01
FWSVD+QLoRA$_{r=8}$	0.63±0.02	0.75±0.01	0.79±0.01	0.81 ±0.01	0.90±0.00	0.92±0.00	**0.55**±0.00	**0.64**±0.03	**0.70**±0.03

Table 4: Average GLUE, CoNLL-2003 and SQuAD v2 results for several compression ranks. The best results for each rank are in **bold**. Extended results are in Tables 6, 8 and 7 in Appendix A.2 for GLUE, CoNLL-2003 and SQuAD v2, respectively.

respectively, other parameters we leave as default.

In summarization experiments, we use `t5-large` model[2], for LoRA we set r to $8, 32$ and s to $16, 64$, respectively.

For MMLU evaluation we use `neural-chat-7b-v3-1`, a checkpoint[3] of `Mistral-7B-v0.1` language model that was fine-tuned on SlimOrca dataset (Mukherjee et al., 2023).

5.3 Compression setups

In this work, we consider two different compression setups depending on the availability of the VRAM in each case. We describe both setups in detail below.

5.3.1 VRAM-unlimited setup

In the first setup, which we refer to as *VRAM-unlimited*, we assume that there is no limitation in terms of the GPU memory and we can train the whole model and collect the gradients for the FWSVD compression. However, we can save some GPU memory by using our FWSVD+LoRA approach without compromising the compression quality after all. We depict this approach in Figure 1. In *VRAM-unlimited* setup, the proposed approach will look as follows:

1. Model is being fine-tuned on the task - it might be the whole model or fine-tuning with LoRA or QLoRA.
2. During fine-tuning, gradients are collected from all fully connected layers for the standard FWVSD approach or low-rank LoRA matrices A and B for our proposed FWSVD+LoRA approach.
3. After fine-tuning, the model is compressed with collected gradients.

[2]huggingface.co/t5-large
[3]huggingface.co/neural-chat-7b-v3-1

In this setup, we do experiments with `roberta-base` model on GLUE, CoNLL-2003 and SQUAD 2.0 benchmarks as this model fits almost any current GPU for full precision training.

5.3.2 VRAM-limited setup

In the second setup, which we refer to as *VRAM-limited*, we cannot collect gradients from the full-sized model because the requirements for a full-sized backward model are beyond our computational capacity. So instead of training LoRA and merging it to our models with compressing, we use pre-trained models on specific tasks and use LoRA training only to collect gradients from the layers for compression. In this case, the experimental design can be described as the following (also shown in Figure 8):

1. We pick an already fine-tuned model for the task.
2. We train LoRA on the same task only to collect gradients, without merging LoRA matrices A, and B back into the model.
3. After training, the fine-tuned model is compressed using collected gradients

With this experiment, we want to confirm whether it is possible to compress a model without re-training it on a concrete task.

5.4 Layers for compression

Since for the majority of the Transformer-based models, the heaviest parts of the model are always the fully-connected layers (Pletenev et al., 2023), we apply LoRA and compress only these parts of the model. For `roberta-base`, we choose the same fully-connected layers: `intermediate` and `output`. For the `t5-large` we compress `wi` and `wo`. Finally, for the `mistral-7b` model we use `gate_proj`, `up_proj` and `down_proj`.

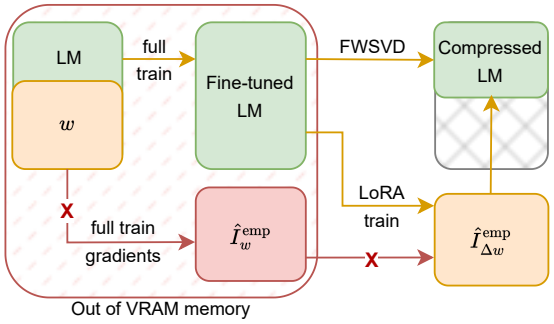

Figure 8: The illustration of the proposed pipeline when the computation of the $\hat{I}_w$ is not possible due to VRAM constraints (denoted with arrows with red crosses), but the computation of $\hat{I}_{\Delta w}$ is possible.

Model Size	60%	69%	79%	90%
Method/Rank	1550	1950	2350	2800
SVD	**0.26**$_{\pm 0.05}$	0.27$_{\pm 0.06}$	0.24$_{\pm 0.04}$	0.24$_{\pm 0.04}$
FWSVD+QLoRA$_{r=16}$	**0.26**$_{\pm 0.05}$	**0.34**$_{\pm 0.06}$	**0.46**$_{\pm 0.11}$	**0.54**$_{\pm 0.12}$
FWSVD+QLoRA$_{r=64}$	**0.26**$_{\pm 0.05}$	**0.34**$_{\pm 0.07}$	**0.46**$_{\pm 0.11}$	**0.54**$_{\pm 0.13}$

Table 5: Average accuracy scores for MMLU benchmark after 5-shot evaluation. The best results for each rank are in **bold**, extended results are in Table 10.

6 Results

In this section, we discuss the performance of our proposed approach compared to simple baseline - SVD and strong baseline - FWSVD on various NLP tasks. Note that we provide the fraction of trainable parameters and peak VRAM utilization for RoBERTa, T5 and Mistral models in Table 1.

6.1 Natural Language Understanding: GLUE

We provide results of our experiments on the GLUE benchmark in Table 4 and Figure 3, extended results are in Appendix A.2 in Table 6. At high compression ranks, neither Fisher Weighted SVD (FWSVD) nor our proposed methods approach the performance of fully fine-tuned models. The difference between FWSVD and our FWSVD+LoRA (QLoRA) approach is minimal, with a mere 4% difference in performance between FWSVD and FWSVD+LoRA$_{r=8}$ at a compression rank of 540. As compression rank decreases, the performance gap between FWSVD and FWSVD+LoRA becomes negligible. However, non-weighted SVD yields significantly inferior performance, comparable to random evaluation. The reported metric represents the average performance across all GLUE tasks. Examination of non-weighted SVD performance reveals that only two subtasks, SST2 and QQP, exhibit quality a little better than random answer, and only under conditions of minimal compression rank 540, compared to FWSVD which shows a superior quality to random answer for most of the subtasks at ranks above 220.

6.2 Named Entity Recognition: CoNLL-2003

We present experimental results on the CoNLL-2003 NER benchmark in Table 4 and Figure 5, extended results are in Table 8 in Appendix A.2.

For the case of CoNLL-2003, we observe a significant difference in compression quality between the standard FWSVD and our proposed FWSVD+LoRA$_{r=8}$ and FWSVD+QLoRA$_{r=8}$ on almost any rank with both FWSVD+LoRA$_{r=8}$ and FWSVD+QLoRA$_{r=8}$ drastically outperforming both FWSVD and SVD by a large margin.

We also note the surprising performance drop for both SVD and FWSVD and analyze it. We suppose that the substantial drop in performance for these approaches between neighbouring ranks (e.g., from 0.8 F_1 at rank 480 to 0.10 F_1 at rank 340 for FWSVD) suggests that the model either ceases functionality due to accumulated error or loses its training knowledge on the Conll-2003 dataset.

6.3 Sequence Tagging: SQuAD 2.0

Similar to GLUE and CoNLL-2003, we experiment with `roberta-base` on the SQuAD v2.0 dataset in a VRAM-unlimited setup. We present experimental results in Table 4 and Figure 6, extended results are in Appendix A.2 in Table 7.

In contrast with GLUE and CoNLL-2003 benchmarks, for SQuAD v2 we observe that for most of the ranks both our proposed FWSVD+LoRA$_{r=8}$ and FWSVD+QLoRA$_{r=8}$ demonstrate superior performance to both FWSVD and SVD across an overwhelming majority of the ranks (see Figure 6).

We suppose that such variability in model performance is because sequence tagging tasks very easily go over-fitting. But in the case of the limitation introduced by LoRA it is more difficult for the model to over-fit and therefore the quality drops less when compressing.

6.4 Text Summarization: XSUM

We experiment with both limited and unlimited VRAM pipelines on XSUM with `t5-large` and discuss the results below.

Experimental results for the unlimited VRAM compression pipeline are presented in Table 2 and Figure 4, extended results are presented in Ta-

ble 9 in Appendix A.2. The summarization quality of the full-sized fine-tuned model is not significantly higher than for those that were fine-tuned with LoRA or QLoRA. Gradually, the quality of all models approximates, but the full-size models remain slightly better. However, as we show in Figure 4, the difference in ROUGE-L is marginal: from 0.2 to 0.5 across all compression ranks, but we save a lot of VRAM by using our proposed FWSVD+LoRA approach 1. Moreover, as we show in Table 9, $r = 8$ is sufficient for both LoRA and QLoRA to demonstrate comparable with the full model fine-tuning performance across all ranks.

We also conduct experiments on XSUM in a *VRAM-limited* pipeline and present results in Table 3 and Figure 7, extended results are in Appendix A.2 in Table 11. For this compression pipeline, we observe a mostly similar trend that was in the *VRAM-unlimited* pipeline with the standard FWSVD being the most effective approach while our FWSVD+LoRA$_{r=8}$ and FWSVD+QLoRA$_{r=8}$ having lower quality - from 2.0 to 0.8 lower ROUGE-L score across all ranks. While the overall performance is comparable to that observed in the *VRAM-unlimited* setup, we note increased variability across different seeds, indicating less stable predictions. The similarity between ranks 8 and 32 for LoRA and QLoRA that was previously observed in the *VRAM-unlimited* pipeline persists in this *VRAM-limited* setting (see Table 11).

6.5 Few-shot Language Understanding: MMLU

In this section, we showcase our approach on the huge LLM Mistral-7B. Namely, we provide results of the experiments on the MMLU benchmark with 5-shot evaluation on Figure 9 and Table 5. Extended results are presented in Appendix A.2 in Table 10. We have no information about VRAM usage from the creators of `neutral-chat-7b-v3-1` and, thus, leave the memory consumption for full model fine-tuning in Figure 9 blank.

In this experiment, we were unable to obtain Fisher weights for the standard FWSVD approach at all. We exceeded NVIDIA A100 GPU memory of 80 GBs and, thus, we do not present results for this approach at all. However, with this experiment, we show that using our proposed VRAM-limited pipeline and QLoRA, it is possible to compress the model with FWSVD+QLoRA.

Despite being worse than the baseline quality of the uncompressed model, both with ranks 16 and 64 our proposed FWSVD+QLoRA provides almost identical performance. For our proposed approach we need only 9.8 GBs of the VRAM compared to 64.82 Gbs of VRAM for SVD: we save 85% of the VRAM. Compression with SVD, however, completely lowers the performance of the model even with the compression to from rank 4096 to 2800.

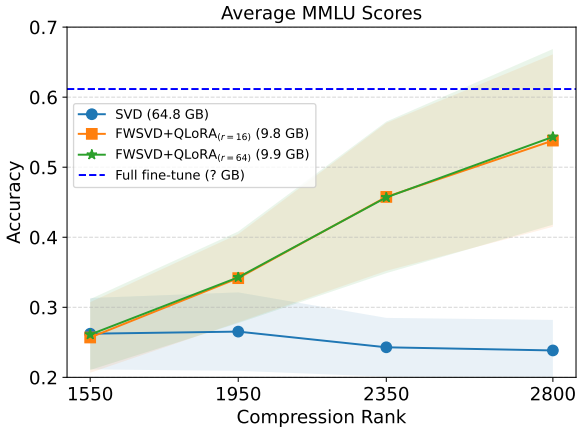

Figure 9: Results for Mistral-7B LLM on MMLU benchmark with 5-shot evaluation. Numerical results are in Tables 5 and 10.

7 Conclusion

In this work, we propose a novel approach combining the state-of-the-art FWSVD language model compression approach and the LoRA PEFT method. We show that to effectively prune the model with FWSVD, one does not require to collect gradients over the whole model. Instead, it is sufficient to collect Fisher information based on layers that we train with LoRA (or QLoRA).

Moreover, we show the effectiveness of our method even in the extremely VRAM-limited setup, where even LoRA-based fine-tuning is not feasible. This is a common used-case when an already pre-trained model, e.g. from Huggingface, is available and its compression it without a huge performance loss is required on a low-end GPU. Using the proposed method, one is able to compress different language models up to 20% without a substantial loss of performance on the downstream tasks, while almost not spending additional VRAM for storing gradients. Our extensive experiments on a five NLP tasks and different types of language models prove the effectiveness of the proposed approach.

8 Limitations

This work has several limitations, which we discuss in this section. This research only specifies fine-tuned tasks such as GLUE, sequence tagging, etc. Maybe this algorithm will not scale to general text prediction tasks. In the case of Mistral 7B, we train only on a small SlimOrca dataset and perhaps the quality of the model will increase if we take a larger dataset. In addition, to compute the Fisher information, we do not completely abandon gradient calculations, although we heavily reduce the requirements, which may also be the subject of future research.

9 Ethics Statement

We propose an approach for computationally efficient fine-tuning and pruning of large neural language models. Both effective fine-tuning and pruning methods reduce the potential carbon footprint and thus the environmental impact.

In the implementation and evaluation of our proposed approach, we use only publicly available code and data to avoid any ethical concerns.

References

Shun-ichi Amari. 1998. Natural gradient works efficiently in learning. *Neural Comput.*, 10(2):251–276.

George C Casella. 2001. *Theory of point estimation*. Springer.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Chun Fan, Jiwei Li, Tianwei Zhang, Xiang Ao, Fei Wu, Yuxian Meng, and Xiaofei Sun. 2021. Layer-wise model pruning based on mutual information. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3079–3090. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Peng Hu, Xi Peng, Hongyuan Zhu, Mohamed M. Sabry Aly, and Jie Lin. 2021. OPQ: compressing deep neural networks with one-shot pruning-quantization. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 7780–7788. AAAI Press.

Ting Hua, Yen-Chang Hsu, Felicity Wang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Numerical optimizations for weighted low-rank estimation on language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 1404–1416. Association for Computational Linguistics.

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

Frederik Kunstner, Philipp Hennig, and Lukas Balles. 2019. Limitations of the empirical fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4158–4169.

Eldar Kurtic, Elias Frantar, and Dan Alistarh. 2023. Ziplm: Hardware-aware structured pruning of language models. *CoRR*, abs/2302.04089.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *NeurIPS*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *CoRR*, abs/2305.11627.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14014–14024.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of GPT-4. *CoRR*, abs/2306.02707.

Kevin P. Murphy. 2012. *Machine learning - a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1797–1807. Association for Computational Linguistics.

Razvan Pascanu and Yoshua Bengio. 2014. Revisiting natural gradient for deep networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Sergey Pletenev, Viktoriia Chekalina, Daniil Moskovskiy, Mikhail Seleznev, Sergey Zagoruyko, and Alexander Panchenko. 2023. A computational study of matrix decomposition methods for compression of pre-trained transformers. In *Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation*, pages 723–742, Hong Kong, China. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*, pages 142–147. ACL.

Nathan Srebro and Tommi S. Jaakkola. 2003. Weighted low-rank approximations. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 720–727. AAAI Press.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5797–5808. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared llama: Accelerating language model pre-training via structured pruning. *CoRR*, abs/2310.06694.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1–9. Association for Computational Linguistics.

Luoming Zhang, Wen Fei, Weijia Wu, Yefei He, Zhenyu Lou, and Hong Zhou. 2023. Dual grained quantization: Efficient fine-grained quantization for LLM. *CoRR*, abs/2310.04836.

A Appendix

A.1 On WNLI exclusion

We intentionally excluded the WNLI task from the GLUE evaluation in this paper for several reasons, and we will describe our motivation below.

First, WNLI is a small dataset compared to others in GLUE. The unreasonably small amount of data may affect the statistical significance of the model performance. In addition, small datasets may not effectively capture the complexity and diversity of language comprehension tasks.

Second, according to the GLUE benchmark description (Wang et al., 2019), there are some "adversarial" examples, "when two examples contain the same sentence, this usually means that they'll have opposite labels".

Since neither uncompressed nor compressed models do not demonstrate an acceptable performance level on WNLI, we exclude this dataset from our GLUE evaluation.

A.2 Extended Results

We provide extended results of our experiments for different LoRA and QLoRA ranks as well as all the compression ranks and standard deviations in Tables 6, 8, 7, 9, 11 and 10.

Method	Rank	CoLA	SST2	MRPC	STS-b	QQP	MNLI	QNLI	RTE
Full FT	768	0.58	**0.94**	**0.90**	**0.90**	**0.90**	**0.88**	**0.92**	**0.72**
LoRA$_{r=1}$		0.55	**0.94**	0.88	**0.90**	0.87	0.86	**0.92**	0.65
LoRA$_{r=8}$		**0.59**	**0.94**	0.89	**0.90**	0.88	0.87	**0.92**	0.71
QLoRA$_{r=1}$		0.56	0.94	0.89	0.89	0.86	0.86	0.92	0.65
QLoRA$_{r=8}$		0.58	0.94	0.90	0.9	0.88	0.87	0.92	0.7
SVD	10	0.01 ± 0.02	0.50 ± 0.03	0.16 ± 0.01	-0.10 ± 0.11	0.46 ± 0.02	0.32 ± 0.00	0.51 ± 0.01	0.48 ± 0.01
FWSVD		-0.01 ± 0.02	$\mathbf{0.57 \pm 0.05}$	0.35 ± 0.33	0.02 ± 0.07	0.33 ± 0.01	0.32 ± 0.01	$\mathbf{0.54 \pm 0.01}$	$\mathbf{0.51 \pm 0.03}$
FWSVD+LoRA$_{r=1}$		$\mathbf{0.02 \pm 0.03}$	$\mathbf{0.57 \pm 0.06}$	0.25 ± 0.04	$\mathbf{0.07 \pm 0.09}$	0.37 ± 0.07	$\mathbf{0.33 \pm 0.01}$	0.52 ± 0.01	0.50 ± 0.05
FWSVD+QLoRA$_{r=1}$		-0.02 ± 0.04	0.55 ± 0.04	$\mathbf{0.66 \pm 0.07}$	-0.01 ± 0.08	0.42 ± 0.07	$\mathbf{0.33 \pm 0.01}$	0.51 ± 0.00	0.48 ± 0.03
FWSVD+LoRA$_{r=8}$		$\mathbf{0.02 \pm 0.02}$	0.53 ± 0.04	0.57 ± 0.15	0.00 ± 0.03	$\mathbf{0.48 \pm 0.03}$	0.32 ± 0.00	0.50 ± 0.01	0.49 ± 0.01
FWSVD+QLoRA$_{r=8}$		-0.00 ± 0.02	0.55 ± 0.05	0.48 ± 0.24	-0.09 ± 0.05	0.42 ± 0.08	$\mathbf{0.33 \pm 0.01}$	0.51 ± 0.00	0.49 ± 0.02
SVD	80	-0.01 ± 0.02	0.51 ± 0.01	0.32 ± 0.19	0.04 ± 0.06	$\mathbf{0.46 \pm 0.00}$	0.32 ± 0.00	0.50 ± 0.01	0.47 ± 0.00
FWSVD		0.01 ± 0.01	$\mathbf{0.65 \pm 0.04}$	0.16 ± 0.00	$\mathbf{0.16 \pm 0.10}$	0.34 ± 0.04	$\mathbf{0.40 \pm 0.03}$	$\mathbf{0.52 \pm 0.01}$	0.47 ± 0.00
FWSVD+LoRA$_{r=1}$		$\mathbf{0.03 \pm 0.05}$	0.53 ± 0.03	$\mathbf{0.61 \pm 0.07}$	0.00 ± 0.05	0.32 ± 0.00	0.34 ± 0.02	0.51 ± 0.00	$\mathbf{0.49 \pm 0.02}$
FWSVD+QLoRA$_{r=1}$		0.01 ± 0.05	0.52 ± 0.03	0.55 ± 0.34	0.02 ± 0.07	0.32 ± 0.00	0.33 ± 0.01	0.51 ± 0.00	0.47 ± 0.00
FWSVD+LoRA$_{r=8}$		0.00 ± 0.00	0.57 ± 0.07	0.31 ± 0.15	0.07 ± 0.14	0.33 ± 0.02	0.34 ± 0.00	0.51 ± 0.00	0.47 ± 0.00
FWSVD+QLoRA$_{r=8}$		0.02 ± 0.04	0.55 ± 0.03	0.45 ± 0.29	0.03 ± 0.04	0.37 ± 0.09	0.37 ± 0.02	0.51 ± 0.00	0.47 ± 0.00
SVD	220	-0.04 ± 0.01	0.54 ± 0.02	0.45 ± 0.25	0.02 ± 0.03	0.45 ± 0.05	0.34 ± 0.01	0.51 ± 0.00	0.48 ± 0.01
FWSVD		$\mathbf{0.04 \pm 0.03}$	$\mathbf{0.76 \pm 0.01}$	0.22 ± 0.08	$\mathbf{0.38 \pm 0.07}$	0.51 ± 0.12	$\mathbf{0.48 \pm 0.01}$	$\mathbf{0.65 \pm 0.06}$	0.49 ± 0.03
FWSVD+LoRA$_{r=1}$		0.01 ± 0.04	0.56 ± 0.02	$\mathbf{0.58 \pm 0.18}$	0.13 ± 0.14	0.45 ± 0.09	0.42 ± 0.02	0.59 ± 0.02	$\mathbf{0.53 \pm 0.01}$
FWSVD+QLoRA$_{r=1}$		0.03 ± 0.05	0.69 ± 0.04	0.37 ± 0.33	0.34 ± 0.05	0.50 ± 0.11	0.45 ± 0.04	0.64 ± 0.01	0.47 ± 0.00
FWSVD+LoRA$_{r=8}$		0.02 ± 0.01	0.67 ± 0.05	0.53 ± 0.30	$\mathbf{0.38 \pm 0.17}$	0.57 ± 0.09	0.44 ± 0.02	0.58 ± 0.07	0.47 ± 0.00
FWSVD+QLoRA$_{r=8}$		0.02 ± 0.03	0.71 ± 0.04	0.46 ± 0.17	0.36 ± 0.09	$\mathbf{0.63 \pm 0.04}$	0.47 ± 0.03	0.58 ± 0.03	0.46 ± 0.01
SVD	340	0.01 ± 0.04	0.54 ± 0.03	0.27 ± 0.17	0.05 ± 0.03	0.51 ± 0.08	0.34 ± 0.00	0.51 ± 0.00	0.48 ± 0.01
FWSVD		$\mathbf{0.16 \pm 0.01}$	$\mathbf{0.86 \pm 0.00}$	$\mathbf{0.77 \pm 0.02}$	$\mathbf{0.73 \pm 0.02}$	$\mathbf{0.78 \pm 0.00}$	$\mathbf{0.65 \pm 0.01}$	$\mathbf{0.82 \pm 0.01}$	$\mathbf{0.57 \pm 0.05}$
FWSVD+LoRA$_{r=1}$		0.11 ± 0.03	0.67 ± 0.11	0.73 ± 0.02	0.34 ± 0.10	0.64 ± 0.05	0.47 ± 0.02	0.74 ± 0.01	0.55 ± 0.02
FWSVD+QLoRA$_{r=1}$		0.19 ± 0.03	0.84 ± 0.01	0.65 ± 0.15	0.65 ± 0.05	0.67 ± 0.01	0.53 ± 0.05	0.69 ± 0.04	0.46 ± 0.01
FWSVD+LoRA$_{r=8}$		0.12 ± 0.00	0.74 ± 0.03	0.75 ± 0.00	0.60 ± 0.12	0.71 ± 0.00	0.55 ± 0.01	0.75 ± 0.01	0.49 ± 0.03
FWSVD+QLoRA$_{r=8}$		0.23 ± 0.00	0.83 ± 0.01	0.75 ± 0.01	0.70 ± 0.04	0.67 ± 0.04	0.57 ± 0.03	0.78 ± 0.02	0.54 ± 0.04
SVD	480	0.02 ± 0.04	0.64 ± 0.02	0.16 ± 0.00	-0.02 ± 0.09	0.58 ± 0.09	0.35 ± 0.01	0.51 ± 0.01	0.47 ± 0.01
FWSVD		0.37 ± 0.02	$\mathbf{0.91 \pm 0.02}$	0.82 ± 0.07	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.86 \pm 0.01}$	$\mathbf{0.82 \pm 0.02}$	$\mathbf{0.89 \pm 0.01}$	0.59 ± 0.10
FWSVD+LoRA$_{r=1}$		0.29 ± 0.01	0.79 ± 0.10	0.76 ± 0.03	0.59 ± 0.11	0.77 ± 0.01	0.65 ± 0.03	0.79 ± 0.01	0.56 ± 0.01
FWSVD+QLoRA$_{r=1}$		0.41 ± 0.02	0.91 ± 0.00	$\mathbf{0.83 \pm 0.03}$	$\mathbf{0.82 \pm 0.01}$	0.74 ± 0.01	0.71 ± 0.04	0.76 ± 0.02	0.60 ± 0.03
FWSVD+LoRA$_{r=8}$		0.32 ± 0.03	0.87 ± 0.01	0.81 ± 0.04	0.78 ± 0.03	0.82 ± 0.02	0.77 ± 0.03	0.84 ± 0.01	0.52 ± 0.03
FWSVD+QLoRA$_{r=8}$		$\mathbf{0.46 \pm 0.00}$	$\mathbf{0.91 \pm 0.00}$	$\mathbf{0.83 \pm 0.03}$	$\mathbf{0.82 \pm 0.01}$	0.79 ± 0.01	0.77 ± 0.01	0.82 ± 0.02	$\mathbf{0.63 \pm 0.02}$
SVD	540	0.02 ± 0.02	0.69 ± 0.04	0.16 ± 0.00	-0.12 ± 0.01	0.65 ± 0.08	0.37 ± 0.02	0.52 ± 0.02	0.47 ± 0.00
FWSVD		0.43 ± 0.19	$\mathbf{0.93 \pm 0.01}$	$\mathbf{0.86 \pm 0.07}$	$\mathbf{0.86 \pm 0.02}$	$\mathbf{0.87 \pm 0.00}$	$\mathbf{0.85 \pm 0.01}$	$\mathbf{0.90 \pm 0.01}$	$\mathbf{0.68 \pm 0.04}$
FWSVD+LoRA$_{r=1}$		0.36 ± 0.03	0.84 ± 0.08	0.79 ± 0.03	0.69 ± 0.05	0.81 ± 0.01	0.72 ± 0.03	0.83 ± 0.00	0.60 ± 0.02
FWSVD+QLoRA$_{r=1}$		0.51 ± 0.02	0.92 ± 0.01	0.85 ± 0.01	0.85 ± 0.00	0.80 ± 0.02	0.79 ± 0.03	0.82 ± 0.01	0.63 ± 0.04
FWSVD+LoRA$_{r=8}$		0.42 ± 0.04	0.91 ± 0.01	0.83 ± 0.03	0.82 ± 0.03	0.83 ± 0.01	0.79 ± 0.00	0.87 ± 0.01	0.58 ± 0.08
FWSVD+QLoRA$_{r=8}$		$\mathbf{0.54 \pm 0.01}$	0.92 ± 0.00	$\mathbf{0.86 \pm 0.02}$	$\mathbf{0.86 \pm 0.01}$	0.84 ± 0.01	0.81 ± 0.01	0.86 ± 0.01	0.65 ± 0.01

Table 6: Extended results on the GLUE benchmark in the *VRAM-unlimited* pipeline. The best results for each rank are in **bold**, and the best results overall are **underlined bold**.

Method	Rank	F1	EM
Full FT	768	**0.82 ± 0.00**	0.79 ± 0.00
LoRA$_{r=1}$		0.80 ± 0.00	0.76 ± 0.00
QLoRA$_{r=1}$		0.80 ± 0.00	0.76 ± 0.00
LoRA$_{r=8}$		0.82 ± 0.00	**0.79 ± 0.00**
QLoRA$_{r=8}$		0.82 ± 0.00	0.78 ± 0.00
SVD	10	0.17 ± 0.02	0.15 ± 0.02
FWSVD		0.03 ± 0.03	0.03 ± 0.03
FWSVD+LoRA$_{r=1}$		0.02 ± 0.01	0.01 ± 0.02
FWSVD+QLoRA$_{r=1}$		0.02 ± 0.01	0.01 ± 0.00
FWSVD+LoRA$_{r=8}$		0.04 ± 0.05	0.04 ± 0.05
FWSVD+QLoRA$_{r=8}$		0.02 ± 0.01	0.01 ± 0.00
SVD	80	0.18 ± 0.10	0.16 ± 0.11
FWSVD		0.49 ± 0.01	0.49 ± 0.01
FWSVD+LoRA$_{r=1}$		0.44 ± 0.08	0.44 ± 0.08
FWSVD+QLoRA$_{r=1}$		0.43 ± 0.17	0.43 ± 0.20
FWSVD+LoRA$_{r=8}$		0.40 ± 0.15	0.40 ± 0.15
FWSVD+QLoRA$_{r=8}$		0.36 ± 0.17	0.35 ± 0.18
SVD	220	0.28 ± 0.04	0.27 ± 0.05
FWSVD		0.48 ± 0.02	0.47 ± 0.02
FWSVD+LoRA$_{r=1}$		0.50 ± 0.00	0.50 ± 0.00
FWSVD+QLoRA$_{r=1}$		0.49 ± 0.10	0.49 ± 0.10
FWSVD+LoRA$_{r=8}$		0.47 ± 0.04	0.47 ± 0.04
FWSVD+QLoRA$_{r=8}$		**0.50 ± 0.00**	**0.50 ± 0.00**
SVD	340	0.30 ± 0.00	0.29 ± 0.01
FWSVD		0.48 ± 0.03	0.45 ± 0.05
FWSVD+LoRA$_{r=1}$		0.50 ± 0.00	0.50 ± 0.00
FWSVD+QLoRA$_{r=1}$		0.50 ± 0.04	0.50 ± 0.04
FWSVD+LoRA$_{r=8}$		0.53 ± 0.01	0.51 ± 0.01
FWSVD+QLoRA$_{r=8}$		**0.55 ± 0.00**	**0.53 ± 0.00**
SVD	480	0.46 ± 0.01	0.45 ± 0.01
FWSVD		0.57 ± 0.02	0.52 ± 0.02
FWSVD+LoRA$_{r=1}$		0.54 ± 0.01	0.52 ± 0.01
FWSVD+QLoRA$_{r=1}$		0.54 ± 0.01	0.53 ± 0.01
FWSVD+LoRA$_{r=8}$		0.59 ± 0.01	0.55 ± 0.01
FWSVD+QLoRA$_{r=8}$		**0.64 ± 0.03**	**0.61 ± 0.03**
SVD	540	0.47 ± 0.02	0.47 ± 0.02
FWSVD		0.64 ± 0.01	0.59 ± 0.01
FWSVD+LoRA$_{r=1}$		0.57 ± 0.02	0.54 ± 0.02
FWSVD+QLoRA$_{r=1}$		0.60 ± 0.01	0.58 ± 0.01
FWSVD+LoRA$_{r=8}$		0.64 ± 0.01	0.61 ± 0.01
FWSVD+QLoRA$_{r=8}$		**0.70 ± 0.03**	**0.67 ± 0.03**

Table 7: Results on SQuADv2 dataset in the *VRAM-unlimited* pipeline. The best results for each rank are in **bold**, and overall best results are **bold underlined**.

Method	Rank	F1	Precision	Recall	Accuracy
Full FT	768	**<u>0.95</u>**	**<u>0.95</u>**	**<u>0.96</u>**	**<u>0.99</u>**
LoRA$_{r=8}$		**<u>0.95</u>**	0.94	0.95	**<u>0.99</u>**
QLoRA$_{r=8}$		0.94	0.93	0.94	**<u>0.99</u>**
SVD	10	0.03 ± 0.02	0.01 ± 0.01	0.12 ± 0.11	0.31 ± 0.45
FWSVD		0.01 ± 0.00	$\mathbf{0.05 \pm 0.05}$	0.00 ± 0.00	$\mathbf{0.81 \pm 0.01}$
FWSVD+LoRA$_{r=8}$		$\mathbf{0.07 \pm 0.02}$	$\mathbf{0.05 \pm 0.01}$	0.15 ± 0.08	0.39 ± 0.17
FWSVD+QLoRA$_{r=8}$		0.04 ± 0.02	0.03 ± 0.01	0.07 ± 0.04	0.11 ± 0.06
SVD	80	0.02 ± 0.02	0.01 ± 0.01	$\mathbf{0.09 \pm 0.15}$	0.30 ± 0.46
FWSVD		0.00 ± 0.00	0.06 ± 0.08	0.00 ± 0.00	0.83 ± 0.00
FWSVD+LoRA$_{r=8}$		$\mathbf{0.03 \pm 0.0}0$	$\mathbf{0.32 \pm 0.02}$	0.02 ± 0.00	$\mathbf{0.84 \pm 0.00}$
FWSVD+QLoRA$_{r=8}$		0.02 ± 0.02	0.21 ± 0.14	0.01 ± 0.01	0.83 ± 0.00
SVD	220	0.01 ± 0.02	0.01 ± 0.01	0.05 ± 0.08	0.62 ± 0.27
FWSVD		0.00 ± 0.00	0.12 ± 0.17	0.00 ± 0.00	0.83 ± 0.00
FWSVD+LoRA$_{r=8}$		$\mathbf{0.47 \pm 0.04}$	$\mathbf{0.61 \pm 0.03}$	0.38 ± 0.04	$\mathbf{0.90 \pm 0.01}$
FWSVD+QLoRA$_{r=8}$		0.39 ± 0.04	0.61 ± 0.05	0.29 ± 0.03	0.89 ± 0.01
SVD	340	0.00 ± 0.00	0.01 ± 0.02	0.00 ± 0.00	0.83 ± 0.00
FWSVD		0.10 ± 0.08	0.45 ± 0.10	0.06 ± 0.05	0.85 ± 0.01
FWSVD+LoRA$_{r=8}$		$\mathbf{0.83 \pm 0.01}$	$\mathbf{0.86 \pm 0.01}$	$\mathbf{0.80 \pm 0.02}$	$\mathbf{0.97 \pm 0.00}$
FWSVD+QLoRA$_{r=8}$		0.81 ± 0.01	0.85 ± 0.01	0.78 ± 0.02	0.96 ± 0.00
SVD	480	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.83 ± 0.00
FWSVD		0.81 ± 0.00	0.87 ± 0.00	0.76 ± 0.05	0.96 ± 0.00
FWSVD+LoRA$_{r=8}$		$\mathbf{0.92 \pm 0.00}$	$\mathbf{0.92 \pm 0.00}$	$\mathbf{0.92 \pm 0.01}$	$\mathbf{0.99 \pm 0.00}$
FWSVD+QLoRA$_{r=8}$		0.90 ± 0.00	0.90 ± 0.00	0.90 ± 0.00	0.98 ± 0.00
SVD	540	0.01 ± 0.01	0.85 ± 0.07	0.00 ± 0.00	0.83 ± 0.00
FWSVD		0.91 ± 0.00	0.92 ± 0.01	0.90 ± 0.00	0.98 ± 0.00
FWSVD+LoRA$_{r=8}$		$\mathbf{0.93 \pm 0.00}$	$\mathbf{0.93 \pm 0.00}$	$\mathbf{0.94 \pm 0.00}$	$\mathbf{0.99 \pm 0.00}$
FWSVD+QLoRA$_{r=8}$		0.92 ± 0.00	0.91 ± 0.00	0.92 ± 0.00	$\mathbf{0.99 \pm 0.00}$

Table 8: Results on CoNLL-2003 dataset in the *VRAM-unlimited* pipeline. The best results for each rank are in **bold**, and overall best results are **<u>underlined bold</u>**.

Method	Rank	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-L-sum
Full FT	768	**41.19**	**17.74**	**33.21**	**33.22**
LoRA$_{r=8}$		38.30	14.87	30.50	30.50
LoRA$_{r=32}$		38.68	15.32	30.95	30.95
SVD	10	**3.56 ± 0.02**	**0.021 ± 0.02**	**3.02 ± 0.20**	**3.15 ± 0.20**
FWSVD		0.78 ± 0.03	0.00 ± 0.01	0.72 ± 0.01	0.723 ± 0.01
FWSVD+LoRA$_{r=8}$		1.57 ± 0.01	0.00 ± 0.00	1.41 ± 0.01	1.44 ± 0.01
FWSVD+QLoRA$_{r=8}$		0.84 ± 0.08	0.00 ± 0.0	0.78 ± 0.08	0.80 ± 0.08
FWSVD+LoRA$_{r=32}$		1.35 ± 0.01	0.00 ± 0.00	1.24 ± 0.01	1.26 ± 0.01
FWSVD+QLoRA$_{r=32}$		0.94 ± 0.10	0.00 ± 0.00	0.86 ± 0.09	0.88 ± 0.09
SVD	100	2.90 ± 0.02	0.04 ± 0.02	2.51 ± 0.03	2.59 ± 0.03
FWSVD		2.31 ± 0.03	0.03 ± 0.01	2.00 ± 0.02	2.03 ± 0.02
FWSVD+LoRA$_{r=8}$		**3.3 ± 0.01**	**0.03 ± 0.00**	**2.87 ± 0.01**	**2.91 ± 0.01**
FWSVD+QLoRA$_{r=8}$		2.34 ± 0.09	0.03 ± 0.00	2.03 ± 0.08	2.06 ± 0.08
FWSVD+LoRA$_{r=32}$		2.83 ± 0.01	0.03 ± 0.00	2.45 ± 0.01	2.49 ± 0.01
FWSVD+QLoRA$_{r=32}$		2.22 ± 0.33	0.03 ± 0.01	1.95 ± 0.26	1.97 ± 0.27
SVD	200	2.20 ± 0.02	0.04 ± 0.01	2.01 ± 0.02	2.03 ± 0.04
FWSVD		4.25 ± 0.02	0.12 ± 0.07	3.92 ± 0.08	3.93 ± 0.02
FWSVD+LoRA$_{r=8}$		**4.49 ± 0.01**	**0.12 ± 0.01**	**4.15 ± 0.02**	**4.15 ± 0.02**
FWSVD+QLoRA$_{r=8}$		3.90 ± 0.24	0.08 ± 0.01	3.63 ± 0.24	3.64 ± 0.24
FWSVD+LoRA$_{r=32}$		4.32 ± 0.01	0.12 ± 0.00	4.00 ± 0.01	4.00 ± 0.01
FWSVD+QLoRA$_{r=32}$		4.33 ± 0.32	0.11 ± 0.02	4.02 ± 0.29	4.03 ± 0.29
SVD	300	5.72 ± 0.02	0.38 ± 0.02	5.05 ± 0.02	5.05 ± 0.02
FWSVD		**14.27 ± 0.04**	**2.37 ± 0.02**	**11.42 ± 0.01**	**11.59 ± 0.01**
FWSVD+LoRA$_{r=8}$		13.31 ± 0.02	2.06 ± 0.02	10.86 ± 0.02	10.9 ± 0.02
FWSVD+QLoRA$_{r=8}$		10.93 ± 1.27	1.48 ± 0.3	9.06 ± 0.91	9.06 ± 0.95
FWSVD+LoRA$_{r=32}$		13.1 ± 0.01	2.13 ± 0.03	10.62 ± 0.01	10.71 ± 0.01
FWSVD+QLoRA$_{r=32}$		12.07 ± 1.04	1.92 ± 0.25	9.80 ± 0.72	9.87 ± 0.76
SVD	400	20.84 ± 0.02	4.69 ± 0.02	16.29 ± 0.02	16.53 ± 0.02
FWSVD		**29.68 ± 0.01**	**8.294 ± 0.01**	**22.56 ± 0.01**	**22.99 ± 0.01**
FWSVD+LoRA$_{r=8}$		29.09 ± 0.01	7.85 ± 0.02	22.23 ± 0.01	22.38 ± 0.01
FWSVD+QLoRA$_{r=8}$		28.20 ± 1.20	7.35 ± 0.69	21.47 ± 0.94	21.59 ± 0.94
FWSVD+LoRA$_{r=32}$		29.57 ± 0.01	8.22 ± 0.02	22.51 ± 0.01	22.81 ± 0.01
FWSVD+QLoRA$_{r=32}$		28.61 ± 0.15	7.81 ± 0.11	21.63 ± 0.10	22.21 ± 0.15
SVD	500	31.94 ± 0.02	10.08 ± 0.02	24.64 ± 0.02	24.83 ± 0.02
FWSVD		**35.37 ± 0.01**	**12.44 ± 0.01**	**27.74 ± 0.01**	**27.84 ± 0.01**
FWSVD+LoRA$_{r=8}$		34.69 ± 0.00	11.84 ± 0.00	27.12 ± 0.00	27.17 ± 0.00
FWSVD+QLoRA$_{r=8}$		34.56 ± 0.51	11.77 ± 0.42	26.96 ± 0.49	27.00 ± 0.49
FWSVD+LoRA$_{r=32}$		34.83 ± 0.00	11.95 ± 0.00	27.14 ± 0.00	27.24 ± 0.00
FWSVD+QLoRA$_{r=32}$		35.04 ± 0.18	12.16 ± 0.08	27.38 ± 0.13	27.53 ± 0.13
SVD	600	36.81 ± 0.02	13.87 ± 0.02	29.23 ± 0.02	29.25 ± 0.02
FWSVD		**38.24 ± 0.00**	**15.00 ± 0.00**	**30.41 ± 0.00**	**30.43 ± 0.00**
FWSVD+LoRA$_{r=8}$		37.75 ± 0.00	14.54 ± 0.00	29.98 ± 0.00	29.99 ± 0.00
FWSVD+QLoRA$_{r=8}$		37.76 ± 0.25	14.57 ± 0.22	29.99 ± 0.22	29.98 ± 0.21
FWSVD+LoRA$_{r=32}$		37.75 ± 0.00	14.54 ± 0.00	29.87 ± 0.00	29.79 ± 0.00
FWSVD+QLoRA$_{r=32}$		38.04 ± 0.12	14.78 ± 0.06	30.24 ± 0.10	30.26 ± 0.11
SVD	700	39.04 ± 0.02	15.88 ± 0.02	31.38 ± 0.02	31.37 ± 0.02
FWSVD		**39.75 ± 0.00**	**16.46 ± 0.00**	**31.81 ± 0.00**	**31.80 ± 0.00**
FWSVD+LoRA$_{r=8}$		39.32 ± 0.00	15.99 ± 0.00	31.38 ± 0.00	31.38 ± 0.00
FWSVD+QLoRA$_{r=8}$		39.37 ± 0.11	16.04 ± 0.14	31.43 ± 0.14	31.42 ± 0.14
FWSVD+LoRA$_{r=32}$		39.36 ± 0.00	16.06 ± 0.00	31.42 ± 0.00	31.42 ± 0.00
FWSVD+QLoRA$_{r=32}$		39.52 ± 0.14	16.22 ± 0.08	31.68 ± 0.14	31.68 ± 0.14

Table 9: Extended results on XSUM dataset for the *VRAM-unlimited* pipeline. The best results for each rank are in **bold**, and overall best results are **underlined bold**.

Method	Rank	Overall	Humanities	Other	Soc. Sciences	STEM
Full FT	4096	**$\underline{0.61 \pm 0.13}$**	**$\underline{0.57 \pm 0.13}$**	**$\underline{0.68 \pm 0.11}$**	**$\underline{0.72 \pm 0.09}$**	**$\underline{0.51 \pm 0.13}$**
SVD	1550	**0.26 ± 0.05**	**0.25 ± 0.03**	0.23 ± 0.05	**0.31 ± 0.05**	0.27 ± 0.06
FWSVD+QLoRA$_{r=16}$		**0.26 ± 0.05**	0.24 ± 0.03	**0.27 ± 0.06**	0.25 ± 0.05	**0.28 ± 0.06**
FWSVD+QLoRA$_{r=64}$		**0.26 ± 0.05**	0.24 ± 0.03	**0.27 ± 0.06**	0.26 ± 0.05	**0.28 ± 0.06**
SVD	1950	0.27 ± 0.06	0.24 ± 0.03	0.26 ± 0.07	0.30 ± 0.06	0.28 ± 0.06
FWSVD+QLoRA$_{r=16}$		**0.34 ± 0.06**	**0.30 ± 0.05**	**0.37 ± 0.07**	**0.40 ± 0.05**	**0.32 ± 0.06**
FWSVD+QLoRA$_{r=64}$		**0.34 ± 0.07**	**0.30 ± 0.05**	**0.37 ± 0.07**	**0.40 ± 0.06**	**0.32 ± 0.07**
SVD	2350	0.24 ± 0.04	0.25 ± 0.03	0.26 ± 0.05	0.22 ± 0.03	0.24 ± 0.05
FWSVD+QLoRA$_{r=16}$		**0.46 ± 0.11**	**0.40 ± 0.10**	**0.53 ± 0.11**	**0.54 ± 0.09**	0.39 ± 0.08
FWSVD+QLoRA$_{r=64}$		**0.46 ± 0.11**	0.39 ± 0.11	**0.53 ± 0.10**	**0.54 ± 0.09**	**0.40 ± 0.08**
SVD	2800	0.24 ± 0.04	0.23 ± 0.03	0.22 ± 0.04	0.24 ± 0.03	0.26 ± 0.06
FWSVD+QLoRA$_{r=16}$		**0.54 ± 0.12**	**0.49 ± 0.13**	**0.60 ± 0.11**	**0.64 ± 0.09**	0.45 ± 0.11
FWSVD+QLoRA$_{r=64}$		**0.54 ± 0.13**	**0.49 ± 0.13**	**0.61 ± 0.11**	**0.64 ± 0.10**	**0.46 ± 0.10**

Table 10: Average accuracy scores for MMLU benchmark after 5-shot evaluation in *VRAM-limited* pipeline. The best results for each rank are in **bold**, and the best overall results are **bold underlined**.

Method	Rank	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-L-sum
Full FT	768	**$\underline{41.19}$**	**$\underline{17.74}$**	**$\underline{33.21}$**	**$\underline{33.22}$**
SVD	10	**3.43 ± 0.11**	**0.02 ± 0.00**	**2.92 ± 0.10**	**3.05 ± 0.09**
FWSVD		0.88 ± 0.20	0.00 ± 0.00	0.80 ± 0.17	0.81 ± 0.17
FWSVD+LoRA$_{r=8}$		1.29 ± 0.29	0.00 ± 0.00	1.18 ± 0.25	1.19 ± 0.26
FWSVD+QLoRA$_{r=8}$		1.41 ± 0.35	0.00 ± 0.0	1.27 ± 0.31	1.29 ± 0.32
SVD	100	2.26 ± 0.14	**0.04 ± 0.01**	1.98 ± 0.12	2.03 ± 0.13
FWSVD		2.33 ± 0.34	0.03 ± 0.00	2.06 ± 0.26	2.08 ± 0.27
FWSVD+LoRA$_{r=8}$		**3.76 ± 0.27**	0.03 ± 0.00	**3.51 ± 0.29**	**3.52 ± 0.29**
FWSVD+QLoRA$_{r=8}$		3.75 ± 0.12	0.03 ± 0.01	3.46 ± 0.06	3.48 ± 0.06
SVD	200	2.32 ± 0.22	0.04 ± 0.01	2.13 ± 0.20	2.15 ± 0.20
FWSVD		4.33 ± 0.24	**0.16 ± 0.01**	3.96 ± 0.22	3.97 ± 0.22
FWSVD+LoRA$_{r=8}$		**5.10 ± 0.52**	0.12 ± 0.01	**4.80 ± 0.39**	**4.79 ± 0.40**
FWSVD+QLoRA$_{r=8}$		4.79 ± 0.26	0.15 ± 0.03	4.49 ± 0.21	4.47 ± 0.21
SVD	300	6.2 ± 0.25	0.42 ± 0.05	5.47 ± 0.24	5.48 ± 0.24
FWSVD		**15.43 ± 0.64**	**2.87 ± 0.19**	**12.10 ± 0.51**	**12.44 ± 0.51**
FWSVD+LoRA$_{r=8}$		13.48 ± 0.84	2.12 ± 0.18	10.73 ± 0.56	10.69 ± 0.60
FWSVD+QLoRA$_{r=8}$		13.78 ± 2.23	2.36 ± 0.61	11.09 ± 1.61	10.95 ± 1.64
SVD	400	22.23 ± 0.49	5.14 ± 0.14	17.35 ± 0.42	17.53 ± 0.39
FWSVD		**30.26 ± 0.19**	**8.81 ± 0.15**	**23.09 ± 0.18**	**23.63 ± 0.15**
FWSVD+LoRA$_{r=8}$		28.92 ± 0.24	7.72 ± 0.20	22.12 ± 0.22	22.22 ± 0.24
FWSVD+QLoRA$_{r=8}$		29.08 ± 0.8	7.83 ± 0.43	22.27 ± 0.64	22.37 ± 0.61
SVD	500	32.14 ± 0.10	10.26 ± 0.04	24.91 ± 0.12	25.03 ± 0.1
FWSVD		**35.61 ± 0.15**	**12.67 ± 0.12**	**27.91 ± 0.16**	**28.07 ± 0.17**
FWSVD+LoRA$_{r=8}$		34.04 ± 0.17	11.12 ± 0.05	26.48 ± 0.09	26.48 ± 0.09
FWSVD+QLoRA$_{r=8}$		33.72 ± 0.40	10.94 ± 0.32	26.25 ± 0.39	26.25 ± 0.40
SVD	600	36.82 ± 0.10	13.79 ± 0.09	29.18 ± 0.13	29.19 ± 0.13
FWSVD		**38.29 ± 0.08**	**15.05 ± 0.03**	**30.52 ± 0.08**	**30.54 ± 0.08**
FWSVD+LoRA$_{r=8}$		36.45 ± 0.08	13.16 ± 0.07	28.73 ± 0.08	28.73 ± 0.08
FWSVD+QLoRA$_{r=8}$		36.22 ± 0.16	12.98 ± 0.15	28.51 ± 0.13	28.5 ± 0.14
SVD	700	38.84 ± 0.07	15.74 ± 0.03	31.18 ± 0.06	31.17 ± 0.07
FWSVD		**39.69 ± 0.01**	**16.39 ± 0.04**	31.86 ± 0.01	**31.86 ± 0.02**
FWSVD+LoRA$_{r=8}$		37.76 ± 0.06	14.31 ± 0.03	29.96 ± 0.06	29.95 ± 0.06
FWSVD+QLoRA$_{r=8}$		37.65 ± 0.10	14.28 ± 0.09	29.85 ± 0.13	29.84 ± 0.13

Table 11: Extended results on XSUM dataset for the *VRAM-limited* pipeline. The best results for each rank are in **bold**, and overall best results are **underlined bold**.