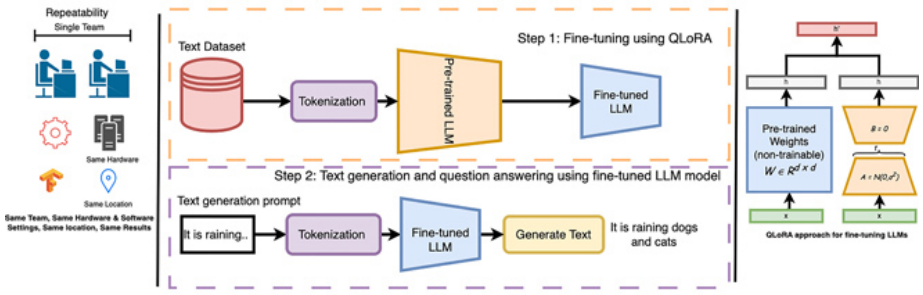
Repeatability of Fine-Tuning Large Language Models Illustrated Using QLoRA

Publisher: IEEE Cite This PDF

Saeed S. Alahmari ; Lawrence O. Hall ; Peter R. Mouton ; Dmitry B. Goldgof **All Authors**

Abstract:
Large language models (LLMs) have shown progress and promise in diverse applications ranging from the medical field to chat bots. Developing LLMs requires a large corpus of data and significant computation resources to achieve efficient learning. Foundation models (in particular LLMs) serve as the basis for fine-tuning on a new corpus of data. Since the original foundation models contain a very large number of parameters, fine-tuning them can be quite challenging. Development of the low-rank adaption technique (LoRA) for fine-tuning, and the quantized version of LoRA, also known as QLoRA, allows for fine-tuning of LLMs on a new smaller corpus of data. This paper focuses on the repeatability of fine-tuning four LLMs using QLoRA. We have fine-tuned them for seven trials each under the same hardware and software settings. We also validated our study for the repeatability (stability) issue by fine-tuning LLMs on two public datasets. For each trial, each LLM was fine-tuned on a subset of the dataset and tested on a holdout test set. Fine-tuning and inference were done on a single GPU. Our study shows that fine-tuning of LLMs with the QLoRA method is not repeatable (not stable), such that different fine-tuned runs result in different performance on the holdout test set.



A visual summary of repeatability task (left), fine-tuning and testing LLMs for text generation, and question answering (middle), and QLoRA fine-tuning approach (right).

Published in: IEEE Access (Volume: 12)

Page(s): 153221 - 153231

DOI: 10.1109/ACCESS.2024.3470850

Date of Publication: 30 September 2024

Publisher: IEEE

Electronic ISSN: 2169-3536

Funding Agency:

SECTION I.
Introduction

Large Language Models (LLMs) developed in the last few years have shown human-level performance for some tasks such as dialogue-based chat bots [1]. The main building block of LLMs is the transformer architecture [2] that enables deep neural networks to learn from sequences of data more efficiently than Long Short-term Memory (LSTM) recurrent neural networks [3]. As a result, many companies have started the race to develop LLMs using a corpus of data freely available on the internet. Some of the developed LLMs hold promise as AI assistants that are able to interact efficiently with humans through dialogue-based chat sessions. Other promising LLMs have been developed for text summarization, translation, and classification such as T5 [4], SwitchTransformers [5], ChatGPT [6], and Llama2 [7]. These LLM models are created with self-supervised training on a data corpus followed by alignment using methods such as Reinforcement Learning with Human Feedback (RLHF) [8].

Fine-tuning LLMs is an efficient approach to improve LLMs' performance [9], [10], [11], especially for datasets of a new domain, or to improve LLMs behaviour or remove undesirable behaviour [11]. A current downside for fine-tuning LLMs is the need for high amounts of memory and computational costs, primari available to large technology companies. To overcome this obstacle, different approaches have been propo to reduce pre-trained model sizes for fine-tuning such as low-rank adaptation (LoRA) [12] and quantized low-rank adaptation (QLoRA) [13]. By modifying the pre-trained model's weight matrix these approaches can effectively reduce memory requirements without a loss of performance.

Repeatability is a critical consideration for maintaining the performance and behaviour of LLM fine-tuning. Repeatability is considered the ability to obtain the same performance after retraining (re-fine-tuning) the model multiple times on same hardware with the same software settings [14], [15].

In this paper, we examine the repeatability of fine-tuning LLMs using the state-of-the-art low rank adaption (i.e. QLoRA). For this purpose, we fine-tuned two LLMs using a quantized low rank adaption approach on a single GPU and report the results of the models on a separate (unseen) held-out test set for each fine-tuned trial. Our contribution can be summarized as follows:

- A study of repeatability using LLM fine-tuning using QLoRA is done for the first time.
- Comparison of repeatability of LLMs was done using four LLMs: GPT-NeoX-20B, GPT-NeoXT-chat-20B, Llama2-7B and Llama2-chat-7B.
- Validation of the repeatability of LLMs using two public datasets.

In the following sections, we provide an overview of related previous studies of repeatability. We discuss the dataset and the four LLMs picked for our study. Furthermore, we explain the QLoRA concept and provide our fine-tuning methodology and results. To the best of our knowledge this is the first direct assessment of the repeatability of fine-tuned LLMs using the QLoRA approach.

SECTION II.
Related Work

Repeatability is a critical consideration for maintaining the performance and behaviour of LLM fine-tuning. Repeatability refers to the ability to obtain the same results by retraining or re-fine tuning of a model by the same team multiple times on the same hardware and software settings [14], [15]. As previously shown, repeatability remains a major challenge for deep learning models [15]. Repeatability requires higher precision computation due to randomization in the deep learning training and fine-tuning such as the GPU operation for enhancing the computation speed (atomic addition operation). Other factors such as weight initialization randomization, order of data by a data loader and regularization methods can effect repeatability of training/fine-tuning of deep learning models. However, the latter randomization can be disabled by seeding or setting determinism options using the deep learning development tools [16], [17], [18].

Prior work has investigated the non-determinism of training deep neural networks and the impact of development tools and the hardware. Zhuang et al. studied the impact of development tools and hardware in the training of deep neural networks determinism [19]. An experimental study by Summers et al. was done where the authors experimentally studied the effect of different parameters and settings on the determinism of training deep neural networks [20]. Such parameters and settings include: weight initialization, data augmentation, data shuffling, and stochastic regularization. The authors proposed two solutions for addressing the non-deterministic training of deep neural networks: accelerated ensemble; and test-time augmentation (TTA). Similarily, non-repeatable (non-deterministic) training of neural networks was studied in [21]. The authors proposed minimum entropy regularizer to increase neural network model confidence. Another approach for reducing the effect of non-repeatable results of training deep neural networks was

proposed by Lemay et al. [22]. This approach uses sampling Monte Carlo dropout predictions at test time to reduce variations in deep neural network predictions. An assessment of repeatability of training deep neural networks was done by Alahmari et al. [15]. After training two deep neural networks for segmentation and classification using the same hardware and software settings for multiple trials, these authors reported non-deterministic results when training deep neural networks.

Dodge et al. studied weight initialization, data order and early stopping impact on fine-tuning BERT on four datasets from the GLUE benchmark [23]. Prior to assessing performance, the authors chose different seeds for weight initialization and data shuffling. Since they allowed randomization in weight initialization and data shuffling, different runs provided different results. Mosbach et al. assessed fine-tuning stability of BERT [24]. Their findings showed that the catastrophic forgetting and fine-tuning in small datasets are not the cause of instability in fine-tuning LLMs [23], [25], [26]. They suggested the use of small learning rates and an increase of number of epochs (iterations) for fine-tuning. However, the results showed a small variance between different runs of fine-tuning, which indicates the fine-tuning of LLMs is not repeatable. Thus, substantial challenges remain with regard to generating repeatable, reproducible, and replicable results w fine-tuning models in deep learning.

SECTION III.
Datasets

The datasets for our experiments are the English quotes dataset and Open-Platypus dataset. The English quotes dataset was originally retrieved from Goodreads quotes1, and can be used for multi-label text classification and text generation [27], [28]. The dataset consists of three columns quote, author, and tags. The quote column contains the quote text in English, and the author column contains the name of the author, and the tags column contains keywords describing the quote. The dataset contains 2508 quotes. We have processed the dataset to remove author and tags columns, while keeping the quote column. Furthermore, we have split the dataset into train (80%) and test (20%) while seeding the split function to ensure the same split is used for each run. The dataset is also publicly available on HuggingFace 2.

The Open-Platypus dataset was used for improving LLM logical reasoning and has been used in developing Platypus2 models [29]. This dataset is comprised of multiple datasets of math and science which have been processed to remove redundancy and similarity using an 80% threshold [30], [31], [32], [33], [34], [35], [36]. This dataset contains 24900 rows which includes instructions and outputs. We have processed the datasets to include the needed tags for each LLM as detailed in Section VII. Furthermore we have split the dataset into train (80%) and test (20%), while seeding the split function to ensure the same split is returned for each fine-tuning trial. The dataset is publicly available on HuggingFace 3.

SECTION IV.
Repeatability

Repeatability is defined as the ability to obtain the same results with a model trained by the same team with the same precision in the same location with the same hardware and software settings [15], [37]. It is worth noting that replicability and reproducibility are not the same as repeatability. Replicability is the ability to obtain exact same results of a model by a different team with different algorithm and different hardware and software setup for multiple trials. Reproducibility refers to obtaining the same exact results with a model trained by a different team in a different location but with the same hardware and software settings for multiple runs. Figure 1 shows a visual illustration of the differences between the repeatability, replicability, and the reproducibility.

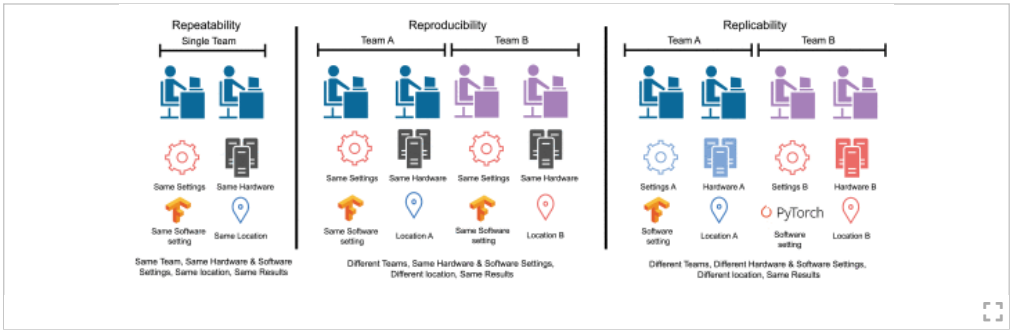


FIGURE 1.
An illustration of the difference between repeatability, replicability, and reproducibility.

SECTION V.
Large Language Models

In this section we describe the large language models (LLM) which we have used for fine-tuning on the datasets described in Section III.

The first LLM we have used is GPT-NeoX-20B from EleutehrAI. This model is a 20 billion parameter transformer autoregressive decoder language model that was pre-trained on the Pile dataset [38]. This LLM was inspired by GPT-3 [1], with some changes such as the use of rotary embeddings [39] instead of learnable positional embedding [40], the use of parallelism in computing the attention and feed-forward layers rather than running them in series, initialization of feed-forward layers was done based on the Wang et al. method [41], and the use of dense layers instead of sparse layers used in GPT-3 to reduce complexity. This LLM is based on 44 layers. The Pile is an open-source English language modeling dataset that contains different high-quality datasets constructed and derived from academic or professional sources. The motivation of using GPT-NeoX-20B is because it is an open sourced autoregressive LLM with code and weights publicly available.

The second model we have used is GPT-NeoXT-chat-20B which is based on fine-tuning the EluteherAI GPT-NeoX-20B model using instructions in dialogue-based interaction. The developer of GPT-NeoXT-chat-20B focused on tasks such as text summarizing, content extraction, question answering, and text classification [42].

The third LLM is Meta Llama-2, which is an updated version of Llama-1 with a number of differences such as training on a new dataset of publicly available data, where the size of the training corpus is 40% larger than the one used for Llama-1 [7]. Furthermore, Llama-2 used a double content length and grouped-query attention (GQA) [43]. Llama-2 comes in different sizes such as Llama2-7B, Llama2-13B, Llama34B, and Llama2-70B which is based on the number of parameters in the LLM (i.e., indicating the size). The training of Llama2 was based on the approach proposed by Touvron et al. [7] with an optimized auto-regressive transformer. The training of Llama-2 was done on 2 trillion tokens of cleaned data, where training used standard transformer architecture [2] with pre-normalization using RMSNorm [44], activation function was SwiGLU [45]. In this paper, we have used Llama2-7B LLM model for assessing repeatability of fine-tuning for text generation using the English quotes dataset.

The fourth LLM we have used in this paper is Llama2-chat-7B by Meta. This model was developed for dialogue-based chatting based on the Llama2 model. This model was tested for safety and showed good performance compared to other closed source models such as ChatGPT and PaLM [7], [46].

SECTION VI.
QLoRA

Both low rank adaption (LoRA) and its extension, quantized low rank adaption (QLoRA) [12], [13], aim to reduce the footprint of GPU memory during fine-tuning. LoRA freezes a pre-trained LLM weight matrix and decomposes the back-propagation update matrix ΔW into A and B using low-rank adaption method. The decomposed weight matrices are small and require less memory, and therefore require smaller GPU memory and less computation during fine-tuning. Figure 2 shows LoRA approach for freezing the pre-trained model and decomposing weight update matrices ΔW into low rank matrices A and B. QLoRA expands LoRA by quantizing the low-rank weight matrices using three components: 4-bits NormalFloat (NF4) data type; double quantization for reducing memory usage during fine-tuning of LLMs; and a paged optimizer for memory spike management and to reduce memory consumption during fine-tuning. QLoRA enables fine-tuning of LLMs on a single GPU of 48GB while maintaining task performance using half-precision (16-bits). Finally, QLoRA freezes and quantizes the weights of the pre-trained model.

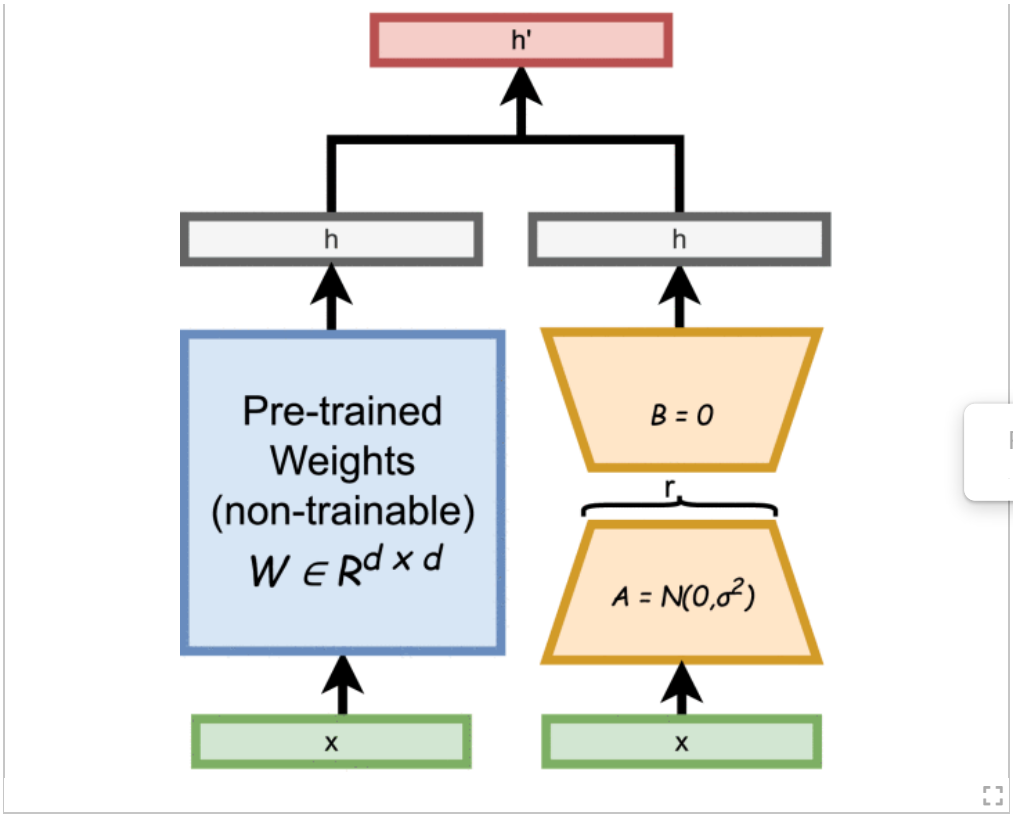


FIGURE 2.
Low rank adaption method, where the pre-trained weights *W* is frozen, and the updated weights matrix is decomposed into two matrices *A* and *B*. The embedding is denoted by h' which is combined from the embedding of the frozen weights *h* and the embedding *h* from the update decomposed weight matrices (*A* and *B*).

SECTION VII.
Experiments

The goal of the present experiment is to assess the repeatability of LLMs fine-tuned with the state-of-the-art QLoRA approach. For this work we used four LLMs, namely, GPT-NeoX-20B and Meta-Llama (Llama-2-7b-hf) for text generation and GPT-NeoXT-chat-20B and Meta-Llama-chat (Llama-2-chat-7b-hf) for dialogue-based question answering. As depicted in Figure 3, our approach consists of two steps: 1) fine-tuning LLMs using the QLoRA approach while saving the fine-tuned model, and 2) text generation using the fine-tuned model.

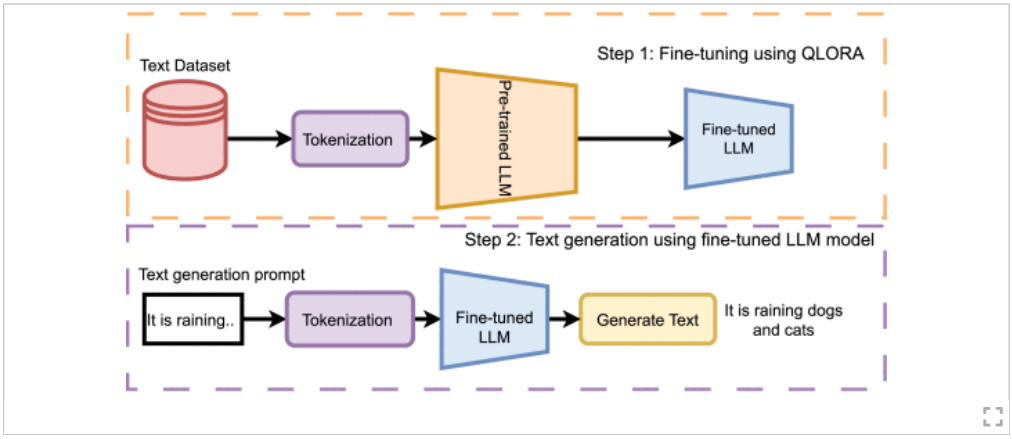


FIGURE 3.
Fine-tuning of LLMs using QLoRA approach, where text dataset is tokenized and used for fine-tuning a pre-trained LLM models (step 1). The fine-tuned model is used for generating text using a text prompt (step2).

Given a pre-trained model W_0 , the goal is to fine-tune the pre-trained model for seven trials t_i where $i \in \{1, 2, 3, 4, 5, 6, 7\}$ under the same hardware and software settings for three epochs. The QLoRA approach freezes the pre-trained model W_0 where $W_0 \in \mathbb{R}^{d \times k}$, and decompose the back-propagation weight matrix ΔW into two matrices A and B, where $A \in \mathbb{R}^{d \times r}$, and $B \in \mathbb{R}^{r \times k}$ with the rank hyper-parameter r. Note, W_0 is frozen and no gradient updates are applied to it, and A and B receive the gradient updates. During a forward-pass the following operation is applied $h = W_0 x + ABx$, where h is the hidden representation and x is the input data. At the end of each fine-tuning trial, the QLoRA fine-tuned adapter weights ΔW_i , where i represents the trial number is saved for testing. During the testing phase, loading the pre-trained model W_0 and the fine-tuned adapter weights ΔW_i is done followed by merging using the Peft software package to form a single model ready for testing using the test set. The merged model W is obtained using $W = W_0 + AB$. The fine-tuning of aforementioned four LLMs was done using a single A100 GPU on the GAIVI cluster at the University of South Florida using two datasets: the English quotes dataset and the Open-Platypus dataset.

Fine-tuning was done on 80% of the dataset, and testing was done on the remaining 20%. The hyper-parameter selection was based on [13]. Although, better results may obtained by tuning the hyper-parameters, our goal in this study is to assess the repeatability of the results rather getting the best results.

We have assessed the effects of two parameters on generated text and answers to questions by LLMs. These two parameters are temperature and nucleus sampling. The default value for these two parameters is 1, and we have set the value to 0.1 to reduce the randomization in text generation. The following two subsections provide the details for fine-tuning each of the four LLMs.

A. GPT-NeoX and GPT-NeoXT-Chat Fine-Tuning

For fine-tuning the GPT-NeoX-20B model on the English quotes dataset (described in Section III), we have downloaded pre-trained model from the Huggingface website using the AutoModelForCausalLm function available in the transformers API. The tokenizer for the input text was also downloaded from the pretrained model using the AutoTokenizer method. For loading the model in a low memory footprint, we have used the QLoRA approach with bits and bytes configurations such as loading the model in 4 bits, and using double quantization, the type of quantization used is normal float 4 bits (NF4), and the computation precision was set to float16. For fine-tuning GPT-NeoX-20B using the English quotes dataset on a GPU, we have used QLoRA, where r was set to 8, lora alpha was set to 32, and lora dropout ratio was set to 0.05. Fine-tuning was done for three epochs with batch size of 1 and learning rate of $2e^{-4}$, the optimizer was paged AdamW. Although, the model was loaded in 4 bits using the QLoRA approach, we have done the fine-tuning in half-precision (i.e., float16). The fine-tuning was done using the transformer trainer API.

GPT-NeoX-20B was fine-tuned for seven runs (trials) for the purpose of evaluating the consistency of the performance of the fine-tuned models on text generation for the test subset. Each trial was fine-tuned independently on the English quotes dataset, with a new loading of the pre-trained model, and setting-up the configurations each time using the same hardware and software. Furthermore, we have studied the impact of the optimizer precision (i.e., paged AdamW 8bits vs. paged AdamW 32 bits). In this context we have fine-tuned seven runs for each of paged AdamW 8bits, and 32 bits optimizer for the purpose of comparing the repeatability of the fine-tuned models. Furthermore, each run was fine-tuned independently on an Nvidia A100 GPU.

For the question-answering repeatability assessment experiment, we have used GPT-NeoXT-chat-20B available on HuggingFace4 using the Open-Platypus dataset. We have downloaded the tokenizer and applied the same parameters that were applied for GPT-NeoX-20B discussed in the last two paragraphs. Furthermore, we have processed the dataset to include the beginning of sentence tag <s> and the end of sentence tag </s>. Moreover, we have added the instruction tag <human> and the response tag <bot>. Fine-tuning of GPT-NeoXT-chat-20B was done for three epochs using an 80% split of Open-Platypus dataset. Fine-tuning was done for seven trials for each of AdamW 8bits and 32bits optimizer. Fine-tuned models for each trial are tested using an unseen 20% of the Open-Platypus dataset.

B. Llama2-7b-Hf and Llama2-Chat-7B Fine-Tuning

For all LLMs we have downloaded the models weights from the Huggingface website using the AutoModelForCausalLM function available from the transformers class. For the Llama2-7B-hf and Llama2-chat-7B-hf model, a user must have an account on Huggingface. Furthermore, a form is required to be completed and submitted to Meta requesting access to Llama models. Then a user can generate a token which is required for authentication when downloading the pre-trained Llama2-7b-hf model. We generated a read token and downloaded Llama2-7b-hf using AutoModelForCausalLM from HuggingFace. We also, downloaded the tokenizer from the pre-trained Llama2-7b-hf using AutoTokenizer. Using the Parameter Efficient Fine-Tuning library (PEFT), we set the configuration for LoRA, where r was set to 8, LoRA Alpha was set to 32. Using the Transformer training arguments setting, the learning rate was set to $1e^{-4}$, the optimizer was set to paged AdamW, learning rate scheduling was set to constant, training epochs was set to

3. The fine-tuning of Llama2-7b-hf was done for three epochs on the training subset of the dataset where the maximum sequence size was set to 512.

We fine-tuned the Llama2-7b-hf pre-trained model for seven runs for each optimizer precision (i.e., paged AdamW 8bits and paged AdamW 32 bits) using the English quotes dataset. For each run, downloading the pre-trained model, and setting-up the configuration of QLoRA LLM fine-tuning under the same hardware and software settings. Furthermore, each run was fine-tuned independently on an Nvidia A100 GPU.

After fine-tuning Llama2-7b-hf on the training subset of the dataset for seven runs per optimization algorithm type, we tested the fine-tuned models on the test subset for the purpose of evaluating the repeatability of the fine-tuning under the same settings.

For question-answering repeatability experiment we have used Llama2-chat-7B model provided by Meta[5]. After downloading the pretrained model, we have processed the dataset (i.e., Open-Platypus) for the form question-answering. For this purposes, we have placed the starting of sentence tag <s> at the beginning o each sentence and </s> at the end of the sentence in the dataset. Furthermore, we have placed the instruc between the two tags <INST> and </INST>. The answer to each instruction comes right after the end of the closing tag of the instruction. All the parameters used for fine-tuning the Llama2-7B model are used for fine-tuning Llama2-chat-7B model as detailed in the previous three paragraphs. Fine-tuning of Llama2-chat-7B was done using the Open-Platypus dataset for seven trials each using two versions of AdamW optimizer: 8bits AdamW and 32bits AdamW. The fine-tuning was done on 80% of Open-Platypus dataset whereas the test was done on 20% of the dataset.

SECTION VIII.
Evaluation

To evaluate the performance of the fine-tuned GPT-NeoX-20B and Llama2-7b-hf, we prompted each fine-tuned model to complete the quotes “Your silence will” and “Let the improvement of yourself keep you”. The response of each fine-tuned model was recorded for comparison. Furthermore, we have used the Perplexity metric for measuring the quality of generated text by calculating the exponentiated average negative of log-likelihood of a tokenized sequence [47]. The perplexity metric is shown in Equation 1.

$$\text{Perplexity}(X) = \exp\left(-\frac{1}{t}\sum_{i}^{t}\log\ p_\theta(x_i|x_{<i})\right) \tag{1}$$

View Source

where, $\log\ p_\theta(x_i|x_{<i})$ is the log likelihood of the current token i conditioned in the preceding tokens $x_{<i}$. It is worth noting that the perplexity metric is computed using the LLM true probability distribution output before sampling or temperature scaling applied. In Table 1, the perplexity of the fine-tuned GPT-NeoX-20B is shown for each run and optimizer setting. As can be observed, the perplexity is different among trials which indicates fine-tuning the LLM using the QLoRA approach is not repeatable. Furthermore, a lower perplexity is obtained by using paged AdamW optimizer 32 bits. Also, the perplexity of the fine-tuned Llama2-7b-hf model is provided for each of seven runs, where in each run training was done twice with two different optimizers. From this table it is clear that the perplexity is different for each run, which is indicated by varying perplexity using paged AdamW optimizer during fine-tuning of Llama2-7b-hf. A different result with small variation in perplexity was observed among the fine-tuned Llama2-7b-hf runs. However, both indicate that fine-tuning using either optimizer, the perplexity and results of Llama2-7b-hf is not repeatable.

TABLE 1 The Perplexity for Fine-Tuned GPT-NeoX-20B and Llama2-7B-hf Models When Tested on an Unseen Subset of the Data (Test Subset). The Perplexity of the Model Trials Fine-Tuned Using the 8bit AdamW Optimizer and the 32bits AdamW Optimizer are Provided

Run #	GPT-NeoX-20B		Llama2-7B-hf	
	8-bit Optim Perplexity	32-bit Optim Perplexity	8-bit Optim Perplexity	32-bit Optim Perplexity
1	21024	8.039	165687.93	3.567
2	3452	8.179	12.471	3.561
3	3028	8.257	109585.75	3.566
4	5440	8.101	14.766	3.566
5	1105	8.391	185572.82	3.541
6	790	8.226	13.541	3.379
7	1780	8.703	233812.79	3.526

In Table 2, we provide the perplexity for different trials (total of seven trials) for each of GPT-NeoXT-chat-20B and Llama2-chat-7B fine-tuning. As observed from the results, the perplexity for AdamW 8-bits optimizer is higher than the perplexity for AdamW 32-bits optimizer which indicates that the AdamW 8-bits optimizer-based fine-tuning is less repeatable than using AdamW 32-bits optimizer.

TABLE 2 The Perplexity of Fine-Tuned GPT-NeoXT-Chat-20B and Llama2-chat-7B Models When Tested on an Unseen Subset of the Dataset (Open-Platypus Dataset) When Using Adamw 8bits and 32bits Optimizer

Run #	GPT-NeoXT-chat-20B Perplexity		Llama2-chat-7b-hf Perplexity	
	8-bit Optim	32-bit Optim	8-bit Optim	32-bit Optim
1	3824	6.531	6.667	5.365
2	6720	6.468	6.490	5.409
3	2400	6.531	145808	5.374
4	3600	6.937	77.938	5.442
5	1752	6.468	46649	5.500
6	14680	6.718	124091	5.398
7	5920	6.562	10769	5.320

In Table 3, we provide a summary of the GPU allocation for our four fine-tuned LLMs when using AdamW 8bits optimizer and AdamW 32bits optimizer. As observed, the GPU memory allocated for AdamW 32bits optimizer is slightly higher than the GPU memory allocated for fine-tuning LLMs using AdamW 8bits optimizer.

TABLE 3 GPU Memory Allocation (%) per LLM Fine-Tuned Using AdamW 8bits and 32bits Optimizer

LLM	AdamW 8bits Optim GPU Memory Allocation (%)	AdamW 32bits Optim GPU Memory Allocation (%)
GPT-NeoX-20B	76.27	76.38
Llama2-7B	17.09	17.12
GPT-NeoXT-chat-20B	47.48	47.50
Llama2-chat-7B	17.11	17.15

In Tables 4 and 5, we show the results of prompting the fine-tuned models using a prompt from the test split of the English quotes dataset, the responses of both GPT-NeoX-20B and Llama2-7b-hf are different for each test. Furthermore, more understandable responses are obtained using fine-tuned LLMs using QLoRA technique with 32bit paged AdamW optimizer. In Tables 6 and 7, we provide the results for prompting GPT-NeoXT-20B and Llama2-7B using a different prompt ("Let the improvement of yourself keep you"). The results show the variations in the response from one fine-tuned model to the other. Moreover, more realistic and understandable responses are observed for fine-tuned models using the Adamw 32bits optimizer.

TABLE 4 The Results of Prompting the Fine-Tuned GPT-NeoX-20B Models With the Prompt: Your Silence Will. The First Column Shows the Number of Runs (i.e., the Number of Fine-Tuning Trials), the Second Column Shows the Results Using the Model Fine-Tuned With the AdamW 8bits Optimizer, and the Third Column Shows the Results Using the Model Fine-Tuned With the AdamW 32bits Optimizer

GPT-NeoX-20B		
Run #	8-bit Optim	32-bit Optim
1	Your silence will amssymb choice..'s.'s.'s.,'s..,'s.	Your silence will not protect you from me. I will not be silent. I will not be quiet.
2	Your silence will AnAn AnAnAnAnAnAnAnAn AnAnAnAnAnAnAnAn AnAn	Your silence will not go with that. You will not be able to say a word. You will be the
3	Your silence will smile smile smile smile smile smile smile smile smile smile smile smile smile smile smile smile smile smile smile	Your silence will not be seen as a lack of it will be much more powerful than any words you can
4	our silence will God God God God God God God God God God God God God God God God God God God	Your silence will not protect you from me. I will not be yours. I will not be yours. I
5	Your silence will.”.”.”.” .”.”.”.”.”.”.”.”.”.”.”.”.”.”	Your silence will not be used as evidence against you.” “ I don't want to talk about it
6	Your silence will is the is the is the is the is the is is is the is the is the is the	Your silence will not protect you. “I don't want to talk about it.
7	Your silence will every every every every every every every every every every every every every every every every every every every	Your silence will not go against you.”””””””””””””””””” ””””

TABLE 5 The Results of Prompting the Fine-Tuned Llama2-7b-hf Models With the Prompt: Your Silence Will. The First Column Shows the Number of Runs (i.e., the Number of Fine-Tuning Trials), the Second Column Shows the Results Using the Model Fine-Tuned With the AdamW 8bits Optimizer, and the Third Column Shows the Results Using the Model Fine-Tuned With the AdamW 32bits Optimizer

Llama2-7b-hf		
Run #	8-bit Optim	32-bit Optim
1	Your silence will ellenśc Multimedia IUraumraum ellenraumisenisenellen IUellenAutresarangre sqlraum Bij	Your silence will not protect you. Your silence will not change the facts
2	Your silence will 100 years later be filled in by the people who love you. You are a part	Your silence will not protect you. You will be found out. Your lies will be uncovered
3	Your silence will tdtdtd tdtdtduwtdtdtdioretdtdt dtdtdtdtd Railtd	Your silence will not protect you. There are things that must be said, no matter the cost.
4	Your silence will 1 day become a language. Your silence will be a song Your silence will be a song Your silence	Your silence will not be tolerated. I will not tolerate your silence
5	Your silence will product releasebah ulosterokrat otra Videoščatu Release Barcel Release Stati Releaseheim	Your silence will not protect you. The first step in forgiveness is the willingness to forgive.
6	Your silence will 100% guarantee you will never be able to tell me anything.” sierpina was a	Your silence will not be heard in heaven. If you keep quiet here
7	Your silence will archivi ĚĚühle konnoreignar chivi konnVorlagearchivi konn oreign konn konnĚ Vorlage	Your silence will not protect you. Your silence will not save you. Your silence will

TABLE 6 The Results of Prompting the Fine-Tuned GPT-NeoX-20B With the Prompt “Let the Improvment of Yourself Keep You”. The First Column Shows the Number of Runs (Fine-Tuned Trials), the Second Column Shows the Results for Fine-Tuning GPT-NeoX-20 Using the AdamW 8bits Optimizer, Whereas the Third Column Shows the Results When Using the AdawW 32bits Optimizer

GPT-NeoX-20B		
Run #	8-bit Optim	32-bit Optim
1	Let the improvement of yourself keep you the the the the the the the the the the the the the the the the the the the the the the	Let the improvement of yourself keep you Always busy with books and a good life is a matter of training, not of force, of
2	Let the improvement of yourself keep you...	Let the improvement of yourself keep you Always busy with books and a good life is a matter of training, not of force, of
3	Let the improvement of yourself keep you...	Let the improvement of yourself keep you so busy at home, and the busy of the day, is the sign of a lazy person
4	Let the improvement of yourself keep you...	Let the improvement of yourself keep you wrapped up for you to do it to others is a very fine art." — — —
5	Let the improvement of yourself keep you...	Let the improvement of yourself keep you Always in the right way, and you Will at last be quite useless." — W
6	Let the improvement of yourself keep you....	Let the improvement of yourself keep you Always in a good mood." I have a friend who is always in a good mood
7	Let the improvement of yourself keep you...	Let the improvement of yourself keep you AlwaysAminAminA

TABLE 7 The Results of Prompting the Fine-Tuned Llama2-7B-hf Models With the Prompt "Let the Improvement of Yourself Keep You". The First Column Shows the Run Number, the Second and Third Columns Show the Results for the Fine-Tuned Models Using AdamW 8bits and 32bits Respectively

Llama2-7b-hf		
Run #	8-bit Optim	32-bit Optim
1	Let the improvement of yourself keep you happy, but always remember that you are a human being, and you have feelings. But you also	Let the improvement of yourself keep you so occupied in doing good, that you shall forget about doing harm.
2	Let the improvement of yourself keep you " "m "m "m "m "m "mommaicm "I am	Let the improvement of yourself keep you Severe, continued, and forever at it. —WILLIAM G.
3	Let the improvement of yourself keep you oreign konnoreigntexttibőltexttoreigniből konn texttiből textt texttxtarttex ttarchivi	Let the improvement of yourself keep you so busy that you have no time to criticize others.
4	Let the improvement of yourself keep you everybody happy." -Friedrich Nietzsche"It's a terrible thing to	Let the improvement of yourself keep you so occupied as to never notice the passing of time. —Marcus A
5	Let the improvement of yourself keep you I have a wonderful friends, but I've been there to see them.""I have been	Let the improvement of yourself keep you SWEETLY BUSY. —JOHN WOODEN
6	Let the improvement of yourself keep you from complaining about the world. 3. Don't worry about what other people think.	Let the improvement of yourself keep you Let the improvement of yourself keep you so occupied that you have no time to criticize others.
7	Let the improvement of yourself keep you everybody else." "You're never too old to set a goal or to dream a	Let the improvement of yourself keep you SWEET FOOLS THAT WE ARE, AS WE G

In Tables 8 and 9, we provide the results of prompting the fine-tuned models: GPT-NeoXT-chat-20B and Llama2-chat-7B using two instructions. Instruction 1 is "If you have no keyboarding skills at all, you will not be able to use a computer. And if you are not able to use a computer, you will not be able to write your essays using a word processing program. If the statements above are true, which one of the following must be true?

- A: If you are not able to write your essays using a word processing program, you have no keyboarding skills.

- B: If you are able to write your essays using a word processing program, you have at least some keyboarding skills.
- C: If you are not able to write your essays using a word processing program, you are not able to use a computer.
- D: If you have some keyboarding skills, you will be able to write your essays using a word processing program."

and instruction 2 is "Tommy: Many people claim that the voting public is unable to evaluate complex campaign issues. The radio advertisements for Peterson in the national campaign, however, discuss complex campaign issues, and Peterson is currently more popular than the other candidates. Jamie: Yes, Peterson is the most popular. However, you are incorrect in claiming that this is a result of Peterson' s discussion of complex campaign issues. Peterson simply strikes the voters as the most competent and trustworthy candidate. Which one of the following, if true, most supports Jamie's counter to Tommy?

- A: Polling data shows that most voters cannot identify Peterson's positions on campaign issues.
- B: Polling data shows that Peterson's present popularity will probably diminish over time.
- C: Peterson's opponents are discussing some of the same issues as Peterson is discussing.
- D: Polling data shows that some voters consider Peterson competent and trustworthy.".

These two instructions were taken from the test split of the Open-Platypus dataset. The results showed the fine-tuning of LLMs for question-answering is not repeatable. More realistic results are observed for fine-tuning LLMs using AdamW 32 bits, however, we provided only the answer letter, and excluded the models' generated text due to the space limitations.

TABLE 8 The Results of Prompting the Fine-Tuned GPT-NeoXT-Chat-20B Models With Two Instructions Provided in Section X. We Show the Output for Each Fine-Tuned Model Using Both AdamW 8bits and 32bits

Run #	8-bits Optim		32-bits Optim	
	Instruction 1	Instruction 2	Instruction 1	Instruction 2
1	„the the,...	„„ the the	D	B
2	the person ..	the person ...	D	A
3	.....	.....	A	C
4	,,,,,,	,,,,,,,	D	A
5	" "	" " >>>	A	A
6	SIOCSIO..	SIOCSIO...	A	A
7	the the the ...	the the the...	D	C

TABLE 9 The Results of Prompting the Fine-Tuned GPT-NeoXT-Chat-20B Models With Two Instructions Provided in Section X. We Show the Output for Each Fine-Tuned Model Using Both AdamW 8bits and 32bits

Run #	8-bits Optim		32-bits Optim	
	Instruction 1	Instruction 2	Instruction 1	Instruction 2
1	D	Yes	B	D
2	C	C	C	D
3	text Void ...	mach "$ "...	D	D
4	If you are still ...	The whole will...	A	D
5	here Answeraban...	answerdiry Mit ...	B	A
6	INSTINST...	INSTINST...	A	B
7	Cpb...	znaj....	D	D

A visual illustration of the disparities between different fine-tuning models in terms of the loss are shown in first and third row of Figure 4 for GPT-NeoX-20B and GPT-NeoXT-chat-20B fine-tuned using the QLoRA approach with 8bits paged AdamW optimizer. Also the loss for fine-tuning GPT-NeoX-20B and GPT-NeoXT-chat-20B using QLoRA with 32bits paged AdamW optimizer are shown. Each sub-figure shows the loss for each run of fine-tuning GPT-NeoX-20b and GPT-NeoXT-chat-20B, and it is clear that each fine-tuned model has a variation in terms of loss.

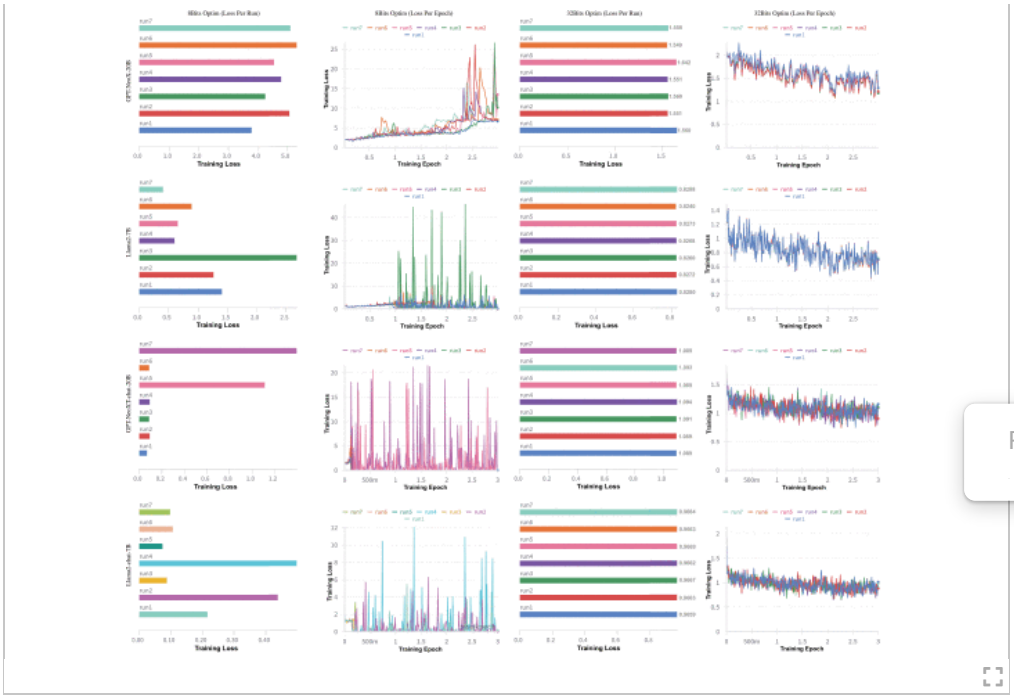


FIGURE 4.
Loss for fine-tuning GPT-NeoX-20B and Llama2-7B. The first row shows the loss for fine-tuning GPT-NeoX-20B, second row shows the loss for fine-tuning Llama2-7B, the third row shows the loss for fine-tuning GPT-NeoXT-chat-20B, and the fourth row shows the loss for fine-tuning Llama2-chat-7B. The first and third column are the loss per run when using 8bits AdamW optimizer, whereas the second and the fourth column show the loss per epoch when using 32bits AdamW optimizer. Due to the small differences between the loss bars in the third column, we have indicated the final loss next to each bar for the LLMs fine-tuned using AdamW 32bits.

Similarly, the loss for fine-tuned Llama2-7B-hf and Llama2-chat-7B-hf models for different runs showed differences from one run to the other for the QLoRA based approach when fine-tuning with two optimizers 8bits and 32bits as shown in Figure 4 second row and fourth row.

A summary of fine-tuning and inference time is shown for each LLM for each run in Tables 10 and 11. A notable observation is the difference in fine-tuning time and inference time among trials. AdamW 32bits optimizer-based fine-tuning requires slightly more training and testing time. However, that is acceptable given the realistic and understandable results fine-tuning LLMs with AdamW 32bits can yield.

TABLE 10 The Fine-Tuned Time and Test Time in Seconds for GPT-NeoX-20B and Llama2-7B Models for Seven Runs

Run #	8-bit Optim				32-bit Optim			
	GPT-NeoX-20B		Llama2-7B		GPT-NeoX-20B		Llama2-7B	
	Train Time (sec)	Test Time (sec)	Train Time (sec)	Test Time (sec)	Train Time (sec)	Test Time (sec)	Train Time (sec)	Test Time (sec)
1	3138.8	3.47	1319.09	1.46	5884.05	2.15	1314.51	1.52
2	3129.58	1.9	1334.6	1.47	5137.61	1.93	1333.23	1.48
3	3132.9	1.96	1332.59	1.48	5140.14	1.93	1319.3	1.46
4	3126.56	1.97	1319.36	1.45	5150.27	1.89	1331.44	1.47
5	3129.38	1.91	1319.69	1.48	5672.52	1.91	1323.11	1.48
6	3134.24	1.93	1326.29	1.48	5924.54	1.92	1322.64	1.51
7	3127.71	1.94	1330.29	1.48	5966.13	1.92	1329.74	1.49

TABLE 11 The Fine-Tuned Time Time and test Time in Seconds for GPT-NeoXT-Chat-20B and Llama2-Chat-7B Models for Seven Runs

Run	8-bit Optim				32-bit Optim			
	GPT-NeoXT-chat-20B		Llama2-chat-7B		GPT-NeoXT-chat-20B		Llama2-chat-7B	
	Train Time (sec)	Test Time (sec)	Train Time (sec)	Test Time (sec)	Train Time (sec)	Test Time (sec)	Train Time (sec)	Test Time (sec)
1	38225.91	9.74	14760.75	7.65	39217.37	9.39	15024.08	7.35
2	38235.36	9.43	14754.61	7.51	39173.43	9.32	15015.84	7.35
3	38266.02	9.57	14596.47	7.4	39244.51	9.35	14979.64	7.46
4	38347.25	9.57	14734.58	7.56	39211.5	9.28	14993.87	7.37
5	38144.76	9.43	14604.02	7.56	39288.29	9.41	14990.42	7.54
6	38276.37	9.35	14702.38	7.49	39182.73	10.02	15017.26	7.49
7	38281.61	7.79	14751.56	7.49	39189.76	7.84	15004.26	7.2

SECTION IX.
Ablation Study

In this ablation study, we have tested the repeatability of the four fine-tuned LLMs when setting the temperature and sampling to a lower value (i.e., temperature = 0.1, and sampling (top-p) = 0.1). The results of setting the temperature and nucleus sampling parameters to a lower value to reduce the randomization in text generation enabled repeatable results when testing the exact same fine-tuned model multiple times. However, non-repeatable results are observed across different independently fine-tuned models. This indicates that fine-tuning with low precision is causing the variation in weights parameters between different fine-tuned models, which consequently caused the differences in the results of the test set.

Furthermore, we have evaluated the pre-trained models (before any fine-tuning) on the test set and report the perplexity score. We found the same per-trained LLM model outputs have the same perplexity for all test trials. The mean perplexity for Llama2-7B-chat-hf for seven test runs is 17.68 ± 0.0 , whereas the mean perplexity for Llama2-7B-hf for seven test runs is 7.04 ± 0.0 . Furthermore, the mean perplexity for GPT-NeoX-20B and GPT-NeoXT-chat-20B for seven test runs are 14.35 ± 0.0 and 16.625 ± 0.0 respectively. These consistent perplexity results are because perplexity is computed using the raw probability distribution before applying the temperature or the sampling technique.

SECTION X.
Discussion

Large language models (LLMs) development and training require large computation resources and memory. However, there is a need to improve LLMs for particular domains such as medical records summarization. The improvement requires fine-tuning pre-trained LLMs with a new and specialized data corpus reflecting the new domain. With the low rank adaptation method (LoRA) and the quantized version of low rank adaptation (QLoRA), fine-tuning can be done on a single GPU. However, many iterations of fine-tuning may be done over time to keep the LLMs updated and to solve privacy, bias, and behaviour concerns. Such fine-tuning can yield different behaviour (sometimes worse than expected) due to the randomization inherent in fine-tuning, rather than the updated data corpus, which leads to increased power-consumption costs, workforce time cost, and delay in availability of trusted services for the end user. Therefore, addressing the repeatability concerns of LLMs is important for quick and interpretable fine-tuning.

Our study focused on LLM fine-tuning using publicly available datasets. We report differences in responses after prompting each fine-tuned model. While using the 32-bits AdamW optimizer showed more realistic responses due to higher precision used during the optimization process, there was nevertheless variation in responses from one fine-tuned LLM trial to another. Furthermore, the perplexity was different from one trial to another for all LLMs. Therefore, it is important to address the repeatability of fine-tuning LLMs using QLoRA and LoRA approaches for obtaining reliable models.

This study focused on assessing the repeatability of fine-tuning LLMs under the same hardware and software settings for multiple trials using QLoRA. For this purpose, we have fine-tuned four LLMs for multiple runs, where each fine-tuning run was done for three epochs. Although, better performance may be obtained by fine-tuning for more epochs, our purpose is to assess repeatability under the same settings rather than finding the optimal results.

Repeatability is a critical consideration for understanding the performance and behavior of LLMs when fine-tuning. Different factors in deep learning training and fine-tuning such as GPU operations and weight initialization can impact repeatability as discussed in Section II. Fine-tuning of LLMs using QLoRA is not

repeatable and leads to inconsistent results, where different runs of fine-tuning yield different outputs and performance. Therefore, fine-tuning LLMs requires attention to the differences in performance that may occur. As observed in this study, the optimizer precision such as paged AdamW with 8-bit and 32-bits precision can result in varying perplexity and model response and as observed from the variation in the loss graph during fine-tuning. Furthermore, fine-tuning time and inference can vary among different trials. Therefore, the observed behavior when fine-tuning LLMs using QLoRA approach highlights the need for stability and interpretability guidelines in deep learning field.

Although, this study focused on the repeatability of LLMs using four foundation models on two tasks text generation and question-answering. There are some limitation to this study including studying repeatability in text classification, content summarization. This study also focused on two open source LLMs, however for our future work, we aim to study repeatability for more large models on different tasks including text classification and content summarization. Furthermore, our future work includes studying the repoducibility and replicability of fine-tuning LLMs.

SECTION XI.
Conclusion

Repeatability of deep learning is important for ensuring stability and interpretability of deployed models. We studied the repeatability of fine-tuning LLMs on the same hardware and software settings for multiple runs. The observed behaviour showed that fine-tuning of LLMs with the QLoRA technique using a single GPU is not repeatable for four LLMs fine-tuned on two publicly available text corpuses. Thus, updating LLMs on a newer corpus of data requires awareness and attention to the differences in the performance that may occur after fine-tuning.

ACKNOWLEDGMENT

Nvidia supported this research with a GPU. The USF Strategic Investment Pool supported this research project.

Authors
Figures
References
Citations
Keywords
Metrics
Footnotes