MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

Parameter Efficient Fine-tuning Mini Project

by Parul Negi & Aseer Ahmad

16 February 2024

Supervisors:

Vedant Nanda
Laurent Bindschaedler
Till Speicher
Krishna P. Gummadi

Problem Introduction

GPT-2

GPT-2 is built on the transformer architecture, which was introduced in the paper "Attention is All You Need" by Vaswani et al. It is known for its large scale. The model has 1.5 billion parameters, making it one of the largest language models at the time of its release. The sheer size contributes to its ability to generate coherent and contextually relevant text.OpenAI released GPT-2 with multiple variations, each differing in the number of parameters:
125M: 125 million parameters.
355M: 355 million parameters.
774M: 774 million parameters.
1.5B: 1.5 billion parameters.
It is pre-trained on a diverse range of internet text, allowing it to capture a broad understanding of language and context. The model learns patterns, structures, and information from a wide variety of sources.
GPT-2 and its variations have contributed significantly to the advancement of natural language processing and have paved the way for even larger and more powerful models like GPT-3. Researchers and developers continue to explore ways to improve and responsibly deploy such models for various applications.

Dataset

databricks-dolly-15k is a corpus of more than 15,000 records generated by thousands of Databricks employees to enable large language models to exhibit the magical interactivity of ChatGPT. Databricks employees were invited to create prompt / response pairs in each of eight different instruction categories, including the seven outlined in the InstructGPT paper, as well as an open-ended free-form category. The contributors were instructed to avoid using information from any source on the web with the exception of Wikipedia (for particular subsets of instruction categories), and explicitly instructed to avoid using generative AI in formulating instructions or responses.

Relevant Tools

Hugging Face Transformers

The transformers library is developed by Hugging Face and is widely used for working with pre-trained language models (such as BERT, GPT, etc.) and training custom models. It provides easy access to a large collection of pre-trained models, tokenizers, and utilities for various NLP tasks.

Torch

PyTorch is an open-source machine learning library developed by Facebook's AI Research lab (FAIR). It is widely used for deep learning and artificial intelligence applications with its key features being Dynamic Computational Graph, Automatic Differentiation , easy of use and pythonic APIs.

Peft

PEFT (Parameter-Efficient Fine-Tuning) is a library for efficiently adapting large pretrained models to various downstream applications without fine-tuning all of a model’s parameters because it is prohibitively costly. PEFT is integrated with the Transformers and we use that along with hugging face trainers.

Training Methodology

Dataset Preparation

The dataset has 4 columns namely ‘instruction', 'context', 'response' and 'category’ . To prepare the dataset we concatenated the following columns in the order :

1. Context
2. Instruction
3. Response

This order of concatenation is only logical to have as any conversation begins with a context followed by an instruction and then finally a response. We would expect our model to infer this behavior only.

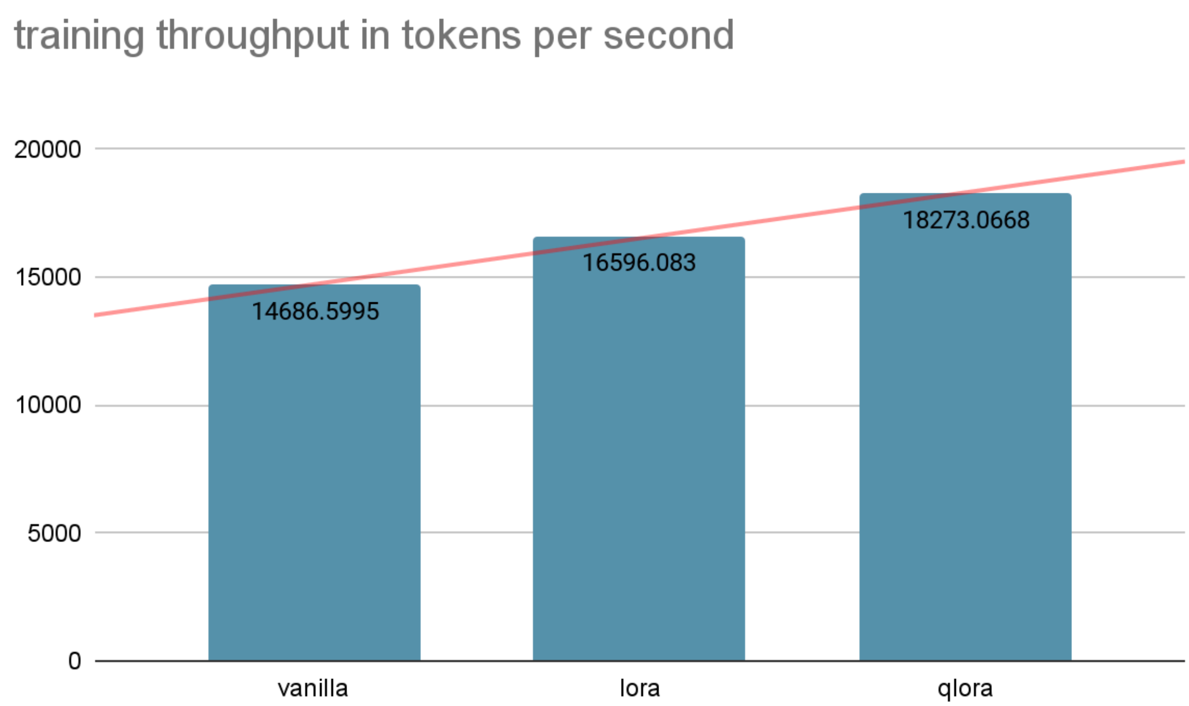
Further , GPT2 Tokenizer was used to prepare the data for model input. We also constrain our input sequence length according to available memory on the computing machine.

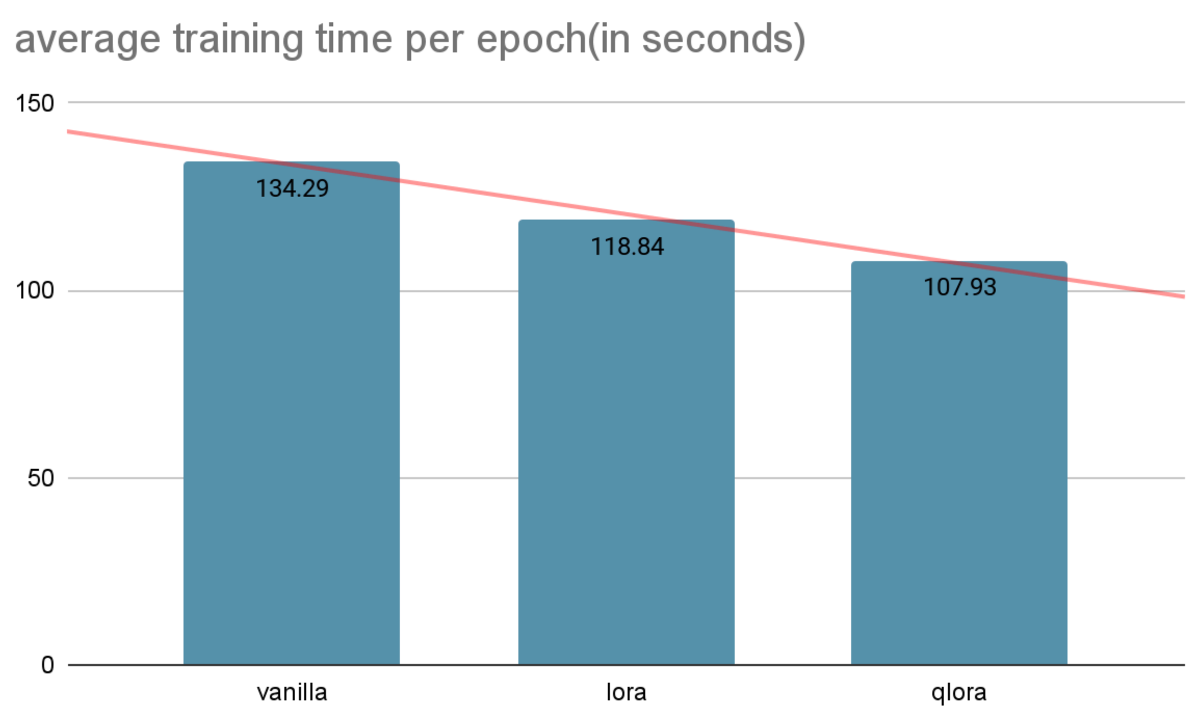
Lora , Qlora & Training

For training with **lora** we use PEFT and further define a lora configuration with rank, alpha and target modules. This defines the hyperparameters for lora and then we prepare the model using this configuration. Trainers of hugging face have been adapted for lora training . So once the target modules are changed training is performed as usual .

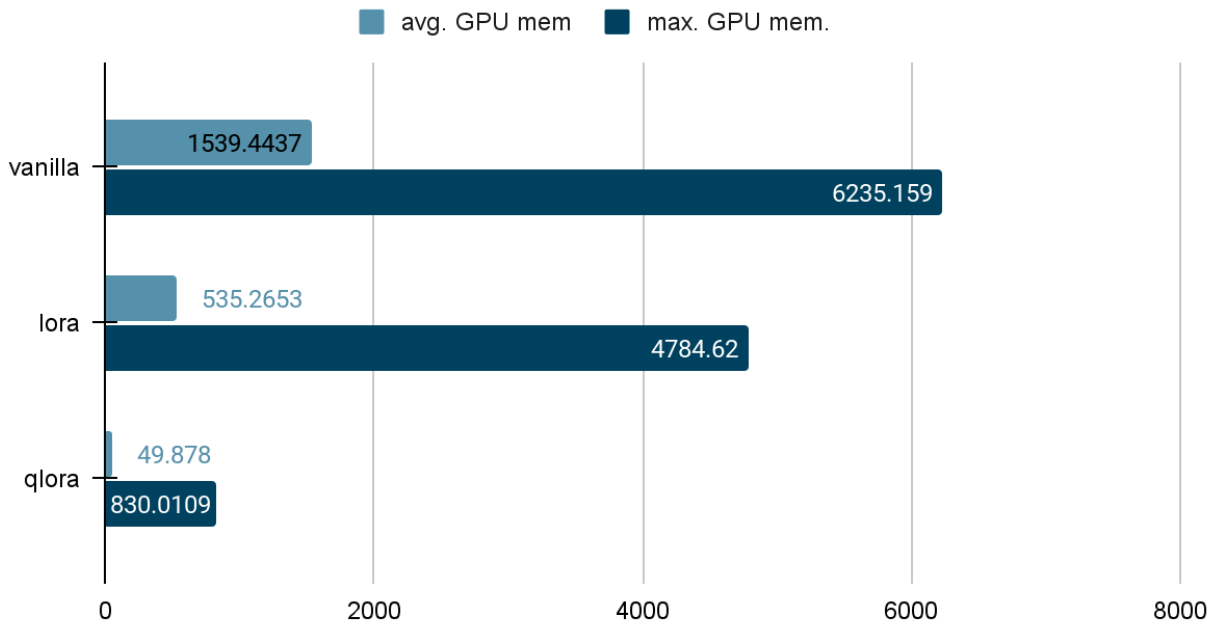
Qlora requires further steps of quantization which we perform using a quantization configuration defined using the BitsAndBytesConfig module. We load the model in this configuration and then once again add lora configuration on top to reduce the parameters for training .

Analysis

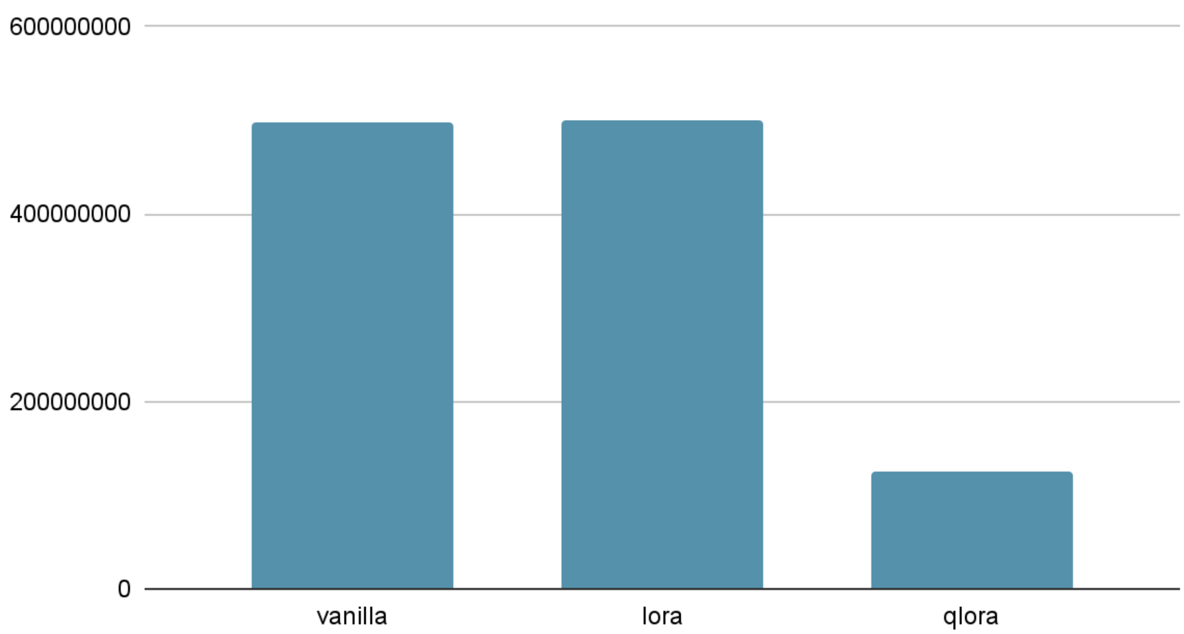

training throughput in tokens per second
20000
15000
10000
5000
0
14686.5995
16596.083
18273.0668
vanilla
lora
qlora

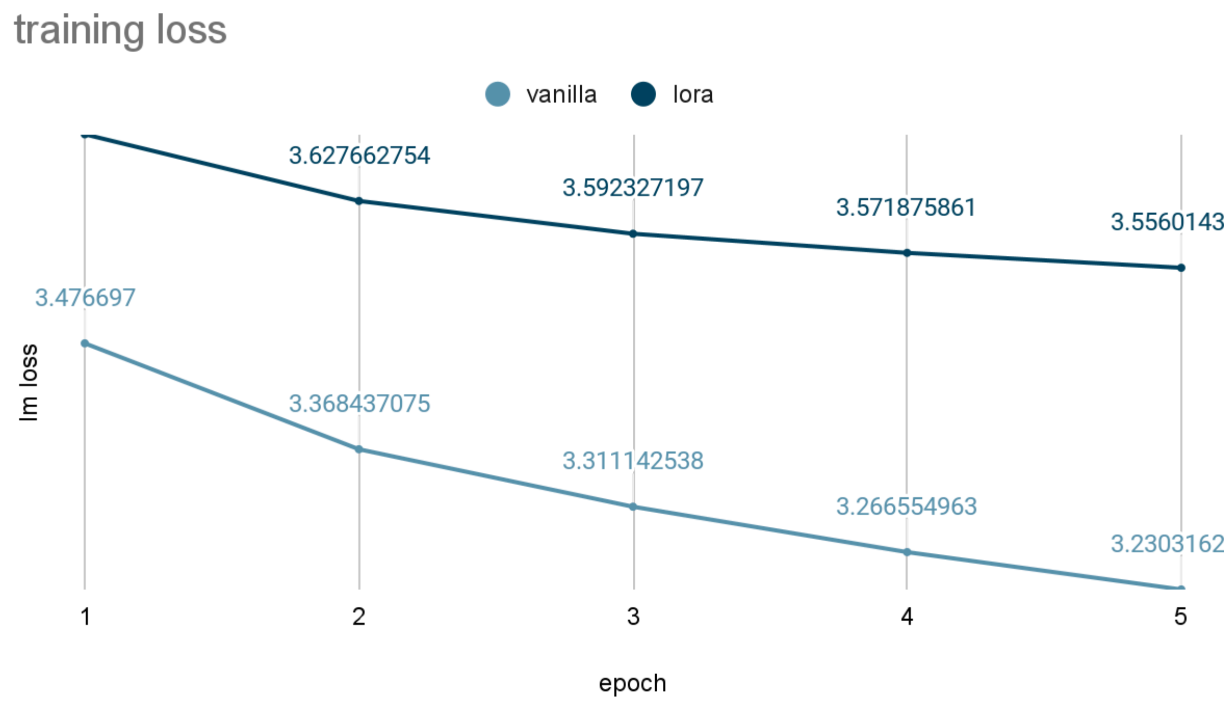

average training time per epoch(in seconds)
150
100
50
0
134.29
118.84
107.93
vanilla
lora
qlora

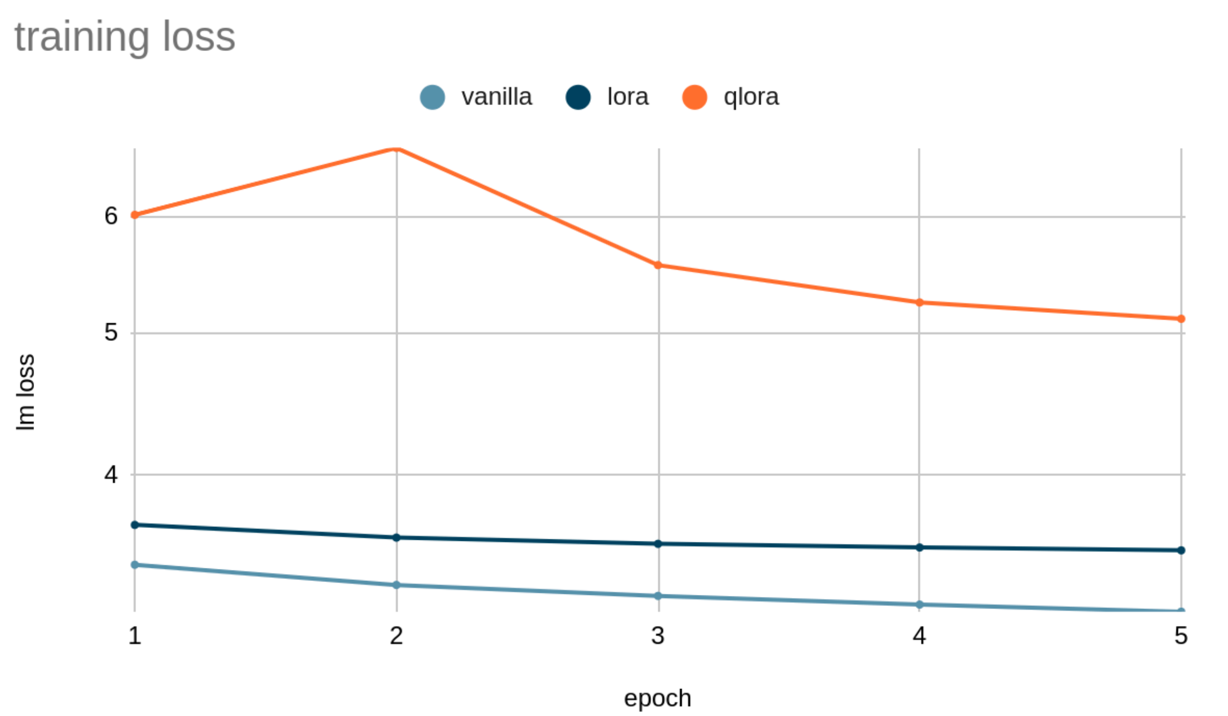
GPU memory(in MB)



on-disk checkpoint size (in bytes)







Results are stored in the folder labeled logs/ with corresponding output files generated from the runs. Config.yaml file contains the key **PEFT_TYPE** which can be set to ‘lora’, ‘qlora’ or empty to run the corresponding experiments. train.py is the main file to start training.