CS772 Project Report
Bayesian Low-Rank Adaptation for Large Language Models

Ujwal Kumar
Roll Number: 201061
Department of Chemical Engineering
Indian Institute of Technology Kanpur
ujwalk20@iitk.ac.in

Harjap Singh
Roll Number: 200405
Department of Chemical Engineering
Indian Institute of Technology Kanpur
harjaps20@iitk.ac.in

Dhwanit Balwani
Roll Number: 200333
Department of Chemical Engineering
Indian Institute of Technology Kanpur
dhwanitb20@iitk.ac.in

Umesh Kumar
Roll Number: 201062
Department of Electrical Engineering
Indian Institute of Technology Kanpur
umeshkumar20@iitk.ac.in

Abstract

Efficient techniques such as low-rank adaptation (LoRA) have revolutionised the fine-tuning of large language models (LLMs), expanding their use to several fields. However, especially after being fine-tuned on tiny datasets, refined LLMs can develop an overconfidence complex. We know Bayesian approaches are very good in estimating uncertainty and can even further be useful for reducing overconfidence and improving calibration. In this work, we use Laplace-LoRA, which treats the LoRA parameters using a Bayesian technique. In particular, Laplace-LoRA significantly enhances the calibration of fine-tuned LLMs by using a Laplace approximation to the posterior across the LoRA parameters. This project is an extension to the previous work (Adam Yang, Maxime Robeyns, 2024).We extend this idea to different LLMs and compare their performances.

1 Introduction

Large Language Models (LLMs) provides a breakthrough in the field of deep learning. LLMs are AI systems capable of understanding, processing and generating data like of human language. These are very large models trained on huge datasets and have billions of parameters like LlaMA-2 has 7 Billion parameters, and Phi-2 has 2.7 Billion parameters. Research is continuously going on reducing the complexity of these LLMs and their parameters and still achieving better results. LLMs are revolutionizing applications in various fields, from chatbots and virtual assistants to content generation, research assistance and language translation. To widen their application to more sectors, fine-tuning LLMs becomes an important part. Fine-tuning helps LLMs to adapt for specific tasks. However a major challenge in fine-tuning is the availability of a limited dataset. LLMs are pre-trained on very huge datasets but fine-tuned on relatively much smaller dataset. This makes fine-tuned LLMs to become overly confident in their predictions, which can be problematic, especially in safety-critical applications like medical diagnosis. So there is need of a solution which can ensure us that their predictions are trustworthy.

Bayesian inference is the famous choice for capturing uncertainty over model parameters and further on predictions. It considers the distribution across parameters rather than just point estimate. Past work (e.g Quiquin Chen,2021) illustrates how incorporating bayesian approach improves calibration

of model and reduce the overconfidence. Limited data availability in fine-tuning exacerbates calibration issues, making Bayesian reasoning more valuable. Applying directly Bayesian approach on parameters can be proven costly as already there are large number of weights.
So to have more efficient approach, parameter efficient fine tunning (PEFT) approach - LoRA has been preferred. It involves fine-tuning only low-rank adapters for each weight matrix and then applying Bayesian inference via Laplace approximation on it. This led to reduction of lot of parameters from billions to just few million LoRA parameters. Then it become easy to apply Laplace approximation on it and hence making it well calibrated.

2 Literature Review

During recent years, many researchers worked on improving callibration of fine-tuned models. Fan et al. (2020) considers prior and approximate posterior over the attention weights rather than over LoRA parameters. Considering models like LlaMA, they have nearly 1 billion attention weights which can result in much memory usage as compare to just considering 6 million LoRA parameters. Some research had been dedicated to regularizing language model fine-tuning to improve calibration, like He et al. (2023) introduced a KL regularization between the output distributions of both fine-tuned and pre-trained language models, and added an L2 regularization to the final embedding vector. But these regularization techniques worked for some specific models like BERT.

The research paper, we have worked upon (Adam Yang, Maxime Robeyns, 2024) consider Laplace inference at all layers to estimate uncertainty over LoRA parameters. They present method, **Laplace LoRA**, which keeps the pretraining and fine-tuning process exactly the same and uses a Laplace approximation to estimate uncertainty around any given mean value of the weights. Laplace LoRA, not only outweigh other methods in computation cost but also in accuracy and reduction in expected calibration error.
We further extend the Laplace-LoRA to new LLMs: LlaMA-3, Phi-2 and Phi-3 and compare the method's performance among these LLMs. We check the reliability of the model's predictions through metrics like ECE and NLL. Further, we compare the metrics of the fine-tuned models with those obtained after Laplace approximation.

3 Background

3.1 LOW-RANK ADAPTATION (LoRA)

LLMs have already pre-trained weight matrix, $W_o \in R^{n_{out} \times n_{in}}$ that are learned during pre-training. Let say we have input **x** and output **y**. Now during LoRA based fine-tuning, a small pertubation ΔW is added to pre-trained weight matrix as:

$$y = (W_o + \Delta W)x = (W_o + \mathbf{BA})x$$

A significant advantage offered by LoRA is that it allows to write pertubation weight matrix as a product of two matrix, **B** $\in R^{n_{out} \times n_{lr}}$ and **A** $\in R^{n_{lr} \times n_{in}}$ as shown above. n_{lr} is much smaller as compare to n_{out} and n_{in}, usually taken to be as 8. Therefore, the total number of LoRA parameters for this weight matrix are $n_{lr}(n_{out}$ + $n_{in})$. These are much smaller number of parameters resulting in significant reduction in memory usage. Ony these parameters are needed to optimize during fine-tuning.

3.2 LAPLACE APPROXIMATIONS

When using Bayesian inference, we search for the posterior distribution of parameters, which provides information about parameter uncertainty and further used to assess posterior predictive distributions. But many times, posterior distribution seems to be intractable. In such cases, one of the efficient technique to find posterior distribution is **laplace approximation** which is basically a gaussian approximation of the posterior distribution. Laplace approximation can be easily derived using Taylor series expansion.

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

Using Laplace Approximation:

$$P(\theta|D) \approx N(\theta|\theta_{MAP}, \Sigma)$$

$$\theta_{MAP} = argmax_{\theta} log(P(D, \theta))$$

To find Σ, following methods can be used:

- Finding the inverse of Hessian matrix: high Computation cost
- Using Diagnol Fisher Information Matrix: assume all weights are independent
- Using GGN approximation: approximation for Hessian matrix
- Using Kronecker-factor method: covariance matrix is approximated as the Kronecker product of two smaller matrices, each capturing the covariance structure within a subset of the parameters

We have used the Kronecker-factor method in Laplace approximations of LoRA adaptors. KFAC, basically approximates the Fisher using blocks for each linear layer and considers the correlation of weights.

4 Methodology

The key steps of our methodology are:

- Incremental computing the low-rank approximation of the Kronecker factors. All the computations are performed using low-rank settings without calculating the full-rank factors.
- The marginal likelihood is optimised using these low-rank approximations.
- Calculating low-rank linearized predictions

4.1 Experimental setup

We fine-tuned LlaMA-2 (meta-llama/Llama-2-7b), LlaMA-3 (meta-llama/Meta-Llama-3-8B), Phi-2 (microsoft/phi-2), Phi-3 (Phi-3-mini-128k-instruct) models leveraging PEFT library to train the LoRA adapters. The adapters were applied to the queries (`q_proj`), values (`v_proj`), and the output layer (`lm_head`). We apply the post-hoc Laplace approximations to the LoRA parameters at the model checkpoints θ_{MAP} obtained from fine-tuning. The LLMs are loaded in 4-bit quantization, which significantly reduces memory and computational requirements while maintaining model performance. The model was trained and evaluated on the WinoGrade (WG) (winogrande) dataset. It is commonly employed as a standard for assessing the performance of natural language understanding models. The dataset is structured as multiple-choice questions. Each question consists of a sentence with a blank (underscore) representing a missing word or phrase. The task is to select the correct option to fill in the blank, based on commonsense reasoning. There are two options A. and B. to choose from. The next token logits corresponding to A/B is selected and the aim is to maximize this likelihood. The prompt used for the dataset as an input to the LLMs:

Prompt: *Select one of the choices that answers the following question:* `{question}` *Choices: A.* `{option1}`. *B* `{option2}`. *Answer:*

The fine-tuning was done using a batch size of 6 for 2500 steps. Checkpoints were saved at a regular interval of 500 gradient steps. At each checkpoint stored the post-hoc Laplace approximation is incorporated and predictions are done using the linearized model as described.

Table 1: Hyperparameters used for fine-tuning LLMs with LoRA adapters

Hyperparameter	Value
lora_r	8
lora_alpha	16
lora_dropout	0.1
Learning Rate (lr)	5e-5
lr_scheduler	Linear
Weight Decay	0

Hyperparameter	Value
train_batch_size	6
eval_batch_size	4
Max sequence length	512
Training Steps	2500
Checkpointing steps	500
Quantization	4-bit

4.2 Metrics

The evaluation is carried out using the following metrics. We use the Negative Log-Likelihood (NLL) and Expected Calibration Error (ECE) metrics to quantify the uncertainty in the predictions of the model.
Negative Log-Likelihood (NLL): It denotes the negative log-likelihood quantifying the difference between the predicted probability distribution and the actual ground truth. It penalizes the incorrect predictions made by the model offering a much nuanced analysis of the model's predictions.

$$\text{NLL} = -\sum_{i=1}^{N} \log P(\hat{y}_i = y_i)$$

Accuracy (ACC): It quantifies the overall correctness of a model's predictions. It is calculated as the ratio of the number of correct predictions to the total number of predictions made by the model.

Expected Calibration Error (ECE): It is a popular measure of model calibration. A model is called calibrated if predicted class probabilities match empirical frequencies. Suppose the predicted probabilities are divided into B bins. Let B_b be the set of samples whose predicted probabilities fall in $I_b = \left(\frac{b-1}{B}, \frac{b}{B}\right]$.

$$f(x) = p(y = c|x)$$

$$\hat{c}_n = max_{c\in\{1,2,...,c\}} f(x_n)$$

$$conf(B_b) = \frac{1}{|B_b|} \sum_{n\in B_b} \hat{c}_n$$

$$acc(B_b) = \frac{1}{|B_b|} \sum_{n\in B_b} I(\hat{y}_n = y_n)$$

$$\text{ECE} = \sum_{m=1}^{B} \frac{|B_m|}{N} |\text{acc}(B_m) - \text{conf}(B_m)|$$

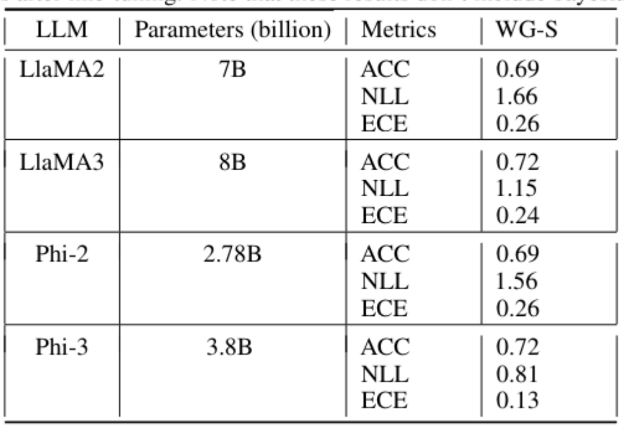
4.3 Code Implementation

We modified the original code implementation tailoring to our needs. The code can be found at https://github.com/ujwalk04/laplace-lora The code implementation of the original paper can be found at https://github.com/MaximeRobeyns/bayesian_lora/.

5 Results

The experimentation took place within a Python 3.10 environment utilizing NVIDIA A100 hardware. Subsequent to 2500 iterations of LoRA fine-tuning, the obtained results are as follows:

Table 2: Results after fine-tuning. Note that these results don't include bayesian intervention.



LLM	Parameters (billion)	Metrics	WG-S
LlaMA2	7B	ACC NLL ECE	0.69 1.66 0.26
LlaMA3	8B	ACC NLL ECE	0.72 1.15 0.24
Phi-2	2.78B	ACC NLL ECE	0.69 1.56 0.26
Phi-3	3.8B	ACC NLL ECE	0.72 0.81 0.13

We evaluate the Laplace-LoRa approach on the checkpoints saved for all the four LLMs. The plots in Figure 1 shows the change in the behaviour of metrics with the gradient steps. We have interpolated the data between two scatter points.

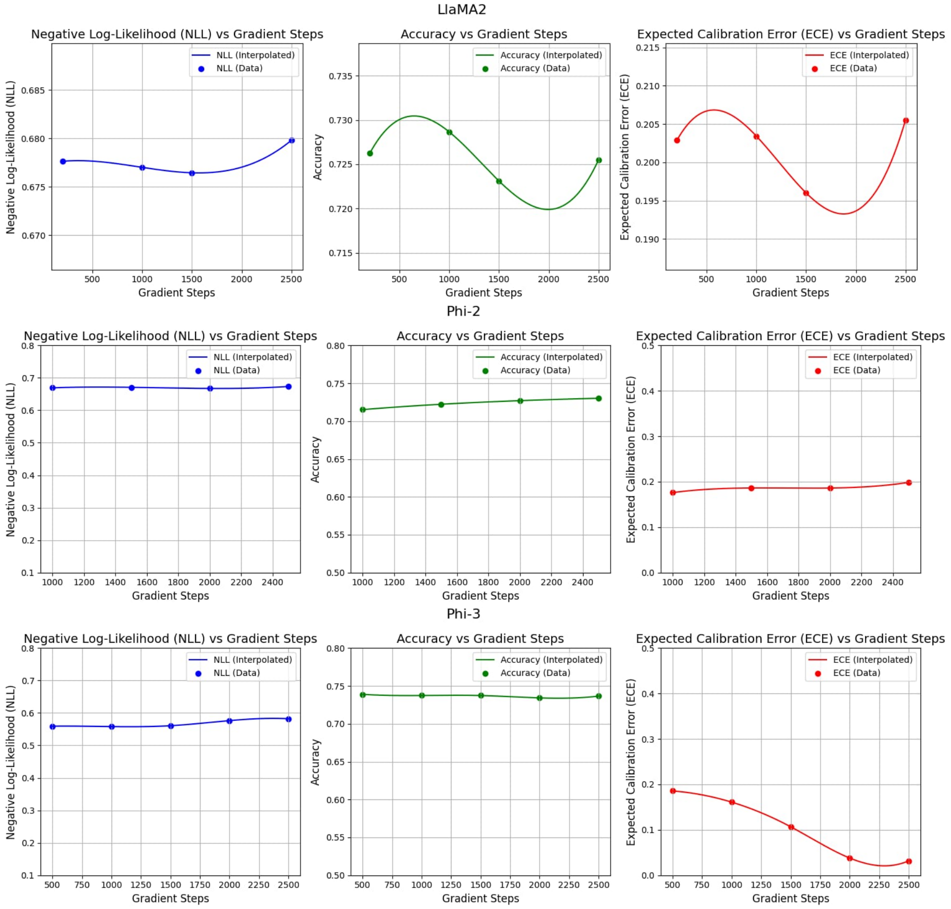
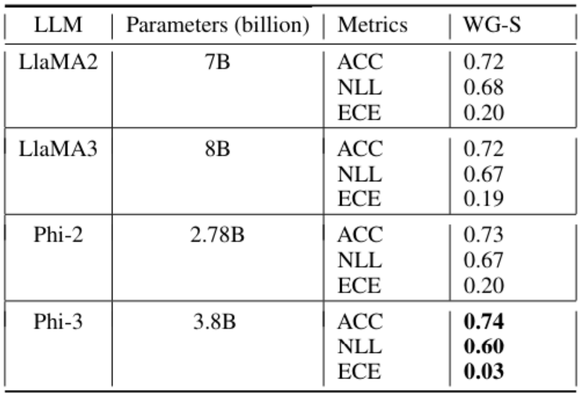


Figure 1: Plots depicting metrics vs number of gradient steps across LLMs. The scatter points are the exact values. Interpolations have been done across the data points.

Post-hoc approximation is performed on the checkpoints saved. All the results obtained are compiled in the table below. It shows a comparative analysis of the performance of the four LLMs: LlaMA-2, LlaMA-3, Phi-2, Phi-3. Phi-3 performs the best among all with the least ECE and NLL indicating more reliable predictions.

Table 3: Comparison of performance of different Large Language Models using using the method of Laplace Approximation on WinoGrande dataset. Results are evaluated after 2500 gradient steps.



LLM	Parameters (billion)	Metrics	WG-S
LlaMA2	7B	ACC NLL ECE	0.72 0.68 0.20
LlaMA3	8B	ACC NLL ECE	0.72 0.67 0.19
Phi-2	2.78B	ACC NLL ECE	0.73 0.67 0.20
Phi-3	3.8B	ACC NLL ECE	**0.74** **0.60** **0.03**

6 Limitations

This project has made advancements to compare the performance of various top-performing open-source LLMs. We evaluated our models on one single dataset. Evaluating the model on multiple other benchmark datasets is crucial for gaining a nuanced understanding of its performance across diverse tasks and data distributions. This would help in gaining confidence in the model predictions across various real-world scenarios. We fine-tuned for 2500 iterations, in contrast to the 10000 iterations in the original paper, due to computational and time constraints. For estimating the model's predictive ability, evaluating it on shifted data distributions also gives us a diverse understanding and helps us gain confidence in the model's predictions.

7 Future Work

The future steps could prioritise the limitations discussed by expanding data diversity, enhancing model training and performing reliability tests. It has also been observed that a simple layer-pruning strategy results in minimal degradation in the model's performance (A Gromov: 2024). We could prune the optimal layers and apply the Laplace-LoRA technique for the remaining layers. This could have complemented other PEFT strategies to further reduce computational resources of finetuning, on the one hand, and can improve the memory and latency of inference on the other hand. Another significant contribution would be in terms of LoRA finetuning. A recent study depicts that LoRA originally introduced, leads to suboptimal finetuning of models with large width (embedding dimension). This suboptimality of LoRA can be corrected simply by setting different learning rates for the LoRA adapter matrices A and B with a well-chosen fixed ratio (S Hayou: 2024). This could be another interesting area which could be utilised to enhance model's performance.

8 Contributions

The work distribution has been indicated below:

Table 1: **Work Distribution**

Name	Work
Ujwal Kumar	Experiments Conducted, Results interpretation, Report making.
Harjap Singh	Literature Review, Code Cleaning, Report making
Dhwanit Balwani	Presentation, Results Analysis, Output compilation
Umesh Kumar	Literature Review, Presentation, Plots and Tables

9 References

[1] Yang, A.X., Robeyns, M., Wang, X., & Aitchison, L. (2024). Bayesian Low-Rank Adaptation for Large Language Models. https://arxiv.org/pdf/2308.13111

[2] Antorán, J., Janz, D., Allingham, J. U., Daxberger, E., Barbano, R. R., Nalisnick, E., & Hernández-Lobato, J. M. (2022). Adapting the linearised Laplace model evidence for modern deep learning. https://arxiv.org/abs/2206.08900

[3] Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). QLoRA: Efficient Finetuning of Quantized LLMs. https://arxiv.org/pdf/2305.14314

[4] Gromov, A., Tirumala, K., Shapourian, H., Glorioso, P. (Year, to be inserted). The Unreasonable Ineffectiveness of the Deeper Layers. https://arxiv.org/pdf/2403.17887

[5] Hayou, S., Ghosh, N., & Yu, B. (2024). LoRA+: Efficient Low Rank Adaptation of Large Models. https://arxiv.org/pdf/2402.12354

[6] Chen, Q., Ni, A., Zhang, C., Wang, J., Xiao, G., & Yu, C. (2021). A Bayesian Neural Network-Based Method to Calibrate Microscopic Traffic Simulators. https://www.hindawi.com/journals/jat/2021/4486149/

[7] Sakaguchi, K., Le Bras, R., Bhagavatula, C., & Choi, Y. (2019). WINOGRANDE: An Adversarial Winograd Schema Challenge at Scale. https://arxiv.org/pdf/1907.10641

[8] Deng, Z., Zhou, F., & Zhu, J. (2022). Accelerated Linearized Laplace Approximation for Bayesian Deep Learning. https://arxiv.org/pdf/2210.12642

[9] Xu, L., Xie, H., Qin, S.-Z. J., Tao, X., Wang, F. L. (Year, to be inserted). Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment. https://arxiv.org/abs/2312.12148

[10] All the course materials covered by Prof. Piyush Rai in the course CS772A: Probabilistic Machine Learning. https://web.cse.iitk.ac.in/users/piyush/courses/pml_spring24/pml.html

[11] Guande He, Jianfei Chen, and Jun Zhu. Preserving pre-trained features helps calibrate fine-tuned language models. https://arxiv.org/abs/2305.19249