Bayesian Hierarchical Low-Rank Adaptation for Multi-Task Emotion Classification in Song Lyrics

Viveka Agrawal[1], **Nicole Liu**[2], **Huizi Xu**[3]
University of California, Irvine[1,2,3]
Irvine, CA 92697
vivekaa@uci.edu
nicolel7@uci.edu
huizix1@uci.edu

Abstract

Fine-tuning large pre-trained language models on multiple downstream tasks often suffers when labeled data are scarce or unevenly distributed. Low-Rank Adaptation (LoRA) alleviates this by injecting light updates into the model, but existing approaches either adapt each task independently or share the same updates across all tasks, both of which can be suboptimal. One proposed solution to this problem is BoRA, a Bayesian hierarchical Low-rank Adaptation framework that places a shared Gaussian prior over the task-specific LoRA weight matrices. A single hyperparameter τ controls the variance around a global mean, creating a mix between fully independent and fully shared adapters for some task or purpose. We evaluate BoRA on an emotion-classification dataset derived from song lyrics (anger, fear, joy, love, sadness, surprise), with training sizes ranging from 50 to 1869 examples. BoRA consistently reduces held-out perplexity: up to around 10% improvement over separate LoRA models and over 90% improvement compared to a single global adapter baseline, demonstrating the critical role of τ in balancing task specialization against knowledge transfer. Our results corroborate with previous work and show that Bayesian hierarchical adaptation offers a way to leverage shared structure across diverse tasks, particularly for ones with little data.

1 Introduction

The usage of large language models (LLMs) has increased in many avenues, among them commercial, academic, and private. LLMs are typically pre-trained on large amounts of data, then finetuned and specialized for a task using more specific data pertaining to that task. However, a problem lies in fine-tuning the model to handle multiple separate tasks. Differing tasks are not best left to the same model, yet training a unique model with new data for each task is time-consuming and computationally expensive. Multiple methods aimed at solving this problem exist, among them merging models or only adjusting a few layers of some training network to handle the individual tasks. This project focuses on an adaptation of the Low-Rank Adaptation (LoRA) method called Bayesian Hierarchical Low-Rank Adaptation (BoRA) [2]. The BoRA approach has the model share information between tasks through a shared Gaussian prior over the task-specific weight matrices from the LoRA method.

This method was originally tested on Norwegian parliamentary speech data by [2], with a unique representative as a task. We will reproduce this method using song lyric data, which has been classified by emotion. Each emotion will represent a task; the emotions and representative tasks are anger, fear, joy, love, sadness, and surprise. BoRA is able to successfully navigate the trade-off between a unique model for each task and a general model for all tasks. More details on LoRA and

Preprint.

BoRA as applied to our project will be presented in Section 2, the data and experimental procedure in Section 3, and results in Section 4.

2 Statistical models and learning algorithms

This section gives greater insight into our model and the underlying algorithms behind it. Much of this is also referenced in [2] and [3].

2.1 Pre-trained autoregressive language model `GPT2`

Our foundational model is `gpt2` [5], a pre-trained autoregressive Transformer language model. Given a sequence of tokens $w_{1:W}$ from a dataset that consists of song lyrics, the model defines the joint distribution as:

$$P(w_{1:W} \mid \boldsymbol{\theta}_{\text{full}}) = \prod_{i=1}^{W-1} \text{GPT2}(w_{i+1} \mid w_{1:i})$$

where $\boldsymbol{\theta}_{\text{full}}$ are the parameters form the pre-trained model and $\text{GPT2}(w_{i+1} \mid w_{1:i})$ denotes the next-token probability given the previous.

2.2 Low-Rank Adaptation (LoRA)

To fine-tune `gpt2` efficiently on multiple song lyric based tasks, we adopted the Low-Rank Adaptation (LoRA) method [3]. LoRA reparameterizes the weight matrices $W \in \mathbb{R}^{n_1 \times n_2}$ into:

$$W = W_{\text{full}} + \frac{\alpha}{r} BA$$

Here, W_{full} is the original weight, and $A \in \mathbb{R}^{r \times n_2}$ and $B \in \mathbb{R}^{n_1 \times r}$ are task-specific trainable matrices. The rank r and scaling factor α are hyperparameters. This model, upon which the rest of this work is based, can be viewed in Figure 1.

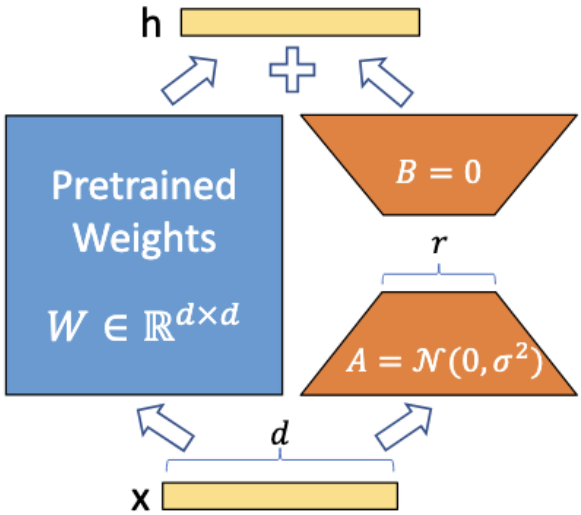

Figure 1: A visualization of the LoRA model, taken from Hu et al[3].

2.3 Bayesian Hierarchical LoRA (BoRA)

We extend LoRA to the multi-task setting by using BoRA [2]. Each task corresponds to a specific emotion from the Spotify lyrics dataset [1]. For each task $d \in \{1, \ldots, D\}$, we learn task-specific LoRA parameters $\boldsymbol{\theta}_d$.

The likelihood over the dataset is:

$$P(D \mid \boldsymbol{\theta}_{1:D}) = \prod_{d=1}^{D} \prod_{n=1}^{N_d} \prod_{i=1}^{W_n - 1} \text{GPT2}(w_{d,n,i+1} \mid w_{d,n,1:i}; \boldsymbol{\theta}_d)$$

We place a Gaussian hierarchical prior over task parameters:

$$\boldsymbol{\theta}_d \sim \mathcal{N}(\boldsymbol{\Theta}, \tau^{-1}\mathbf{I})$$

with $\boldsymbol{\Theta}$ being the global mean of the adapter weights and τ controlling the degree of information sharing. The posterior becomes:

$$P(\boldsymbol{\theta}_{1:D} \mid D, \tau) \propto \prod_{d=1}^{D} P(D_d \mid \boldsymbol{\theta}_d) \cdot \mathcal{N}(\boldsymbol{\theta}_d \mid \boldsymbol{\Theta}, \tau^{-1}\mathbf{I})$$

Here, τ also helps control how similar the task parameters $\boldsymbol{\theta}_d$ are to the hierarchical mean parameters Θ. As τ increases, the task parameters are considered to be more similar to one another, indicating one overall model for the individual tasks. By contrast, a smaller τ indicated greater variation in the task parameters, allowing each task to be treated as an individual model.

2.4 Optimization

We perform a MAP estimation of the task-specific parameters $\boldsymbol{\theta}_{1:D}$ and shared global prior $\boldsymbol{\Theta}$ using the AdamW optimizer [4]. The objective function is:

$$\mathcal{L}(\boldsymbol{\theta}_{1:D}, \boldsymbol{\Theta}) = -\sum_{d=1}^{D} \log P(D_d \mid \boldsymbol{\theta}_d) + \frac{\tau}{2} \sum_{d=1}^{D} \|\boldsymbol{\theta}_d - \boldsymbol{\Theta}\|^2$$

We adjust the learning rate proportionally to τ to maintain stability across different prior strengths.

2.5 Evaluation Metric

Model performance is evaluated using perplexity, a standard metric in language modeling. Perplexity is calculated as the exponentiated average negative log-likelihood per token:

$$\text{Perplexity} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N} \log P(w_i \mid \text{context})\right)$$

Lower perplexity indicates a better fit to the lyric data across tasks.

3 Data and experiment

3.1 Data

Our dataset was hosted on Kaggle and includes data about song lyrics, the artist, and primary emotion, amongst other data [1]. This dataset originally had around 500,000 data entries, which was narrowed down to 4933 entries after data cleaning and taking into consideration computational resources. Songs that belonged to categories with under 5000 entries were first removed from the original dataset, then a random selection of songs were selected from the dataset for each task/emotion while still preserving the original ratio of songs per emotion. The emotion `surprise` had the lowest number of songs in the final working dataset at 50 entries and `joy` the most at 1869 entries. This dataset was further split into training, validation, and testing sets. The amount of training data per task can be seen in Figure 2.

3.2 Experiment

The experiment was finally carried out by adapting the BoRA code used by Eide and Frigessi [2], replacing their use of the language model `opt-350m` from Facebook with `gpt2` [5] from OpenAI.

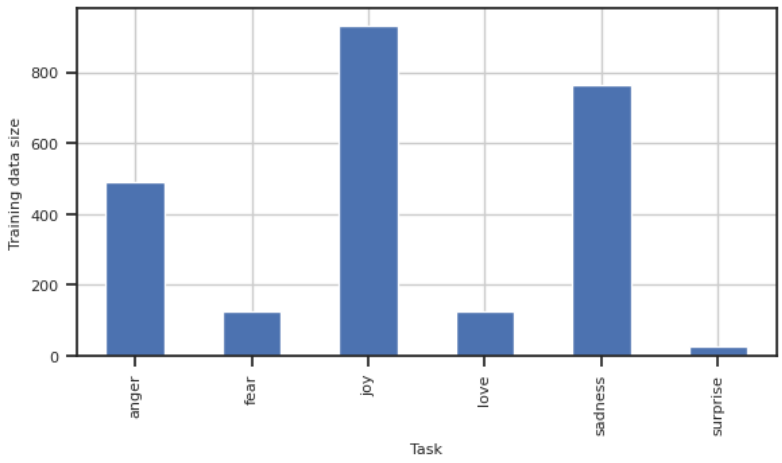

Figure 2: Amount of training data per task.

The model was rerun several times with different learning rates and τ parameters to control the information sharing between tasks, the values of which are included in Table 1. Each iteration was allowed to run for a maximum of 20 epochs, with an early stopping point after 5 epochs if the validation log-likelihood did not improve.

These models were trained through the usage of Google Colab, limiting both the amount of training data used and our pre-trained language model options.

4 Results

After testing several learning rate and τ hyperparameter combinations, the one that performed best in terms of validation perplexity was a learning rate of 0.001 and a τ hyperparameter of 100. This hyperparameter strikes a good balance between treating tasks as one, with one underlying model, and treating the tasks as individual, with one model per task. The validation perplexities for each combination of learning rate and tau are in Table 1.

LR	τ	Validation Perplexity
10^{-4}	0	29.63
10^{-5}	10^{0}	34.90
10^{-4}	10^{1}	28.74
10^{-3}	10^{2}	**27.78**
10^{-2}	10^{3}	29.06
10^{-1}	10^{4}	652.66

Table 1: Learning rates, τ, and validation perplexities.

Additionally, looking at the log-likelihood for a learning rate of 0.001 and a τ hyperparameter of 100 reveals an increasing trend as the number of epochs increases, indicating the model is learning appropriately (see Figure 3).

We can also compare the relative improvement between each model, comparing the lowest validation perplexity model, the individual models approach ($\tau = 0$), and the single model approach ($\tau = 10000$). Our final model improves upon the one model approach for all tasks approach by over 90%, while the single models per task improvement varies around 10%. The relative difference in perplexity is calculated from the validation perplexity scores.

The final perplexity per task for the testing set, as well as the overall perplexity, likelihood, and loss are included in Table 2. The final testing perplexity has a slight improvement over the validation perplexity.

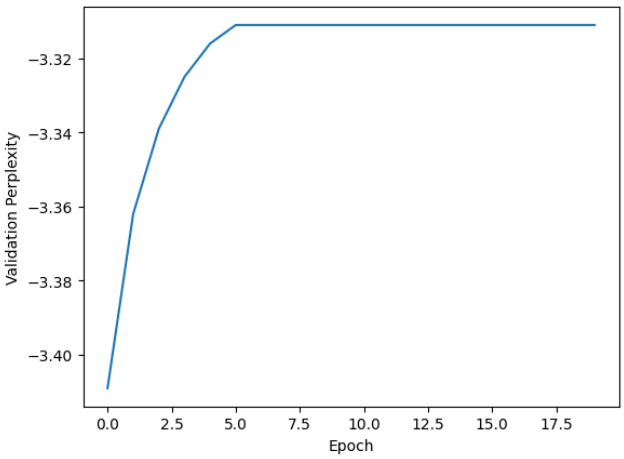

Figure 3: Increasing validation log-likelihood for learning rate 0.001 and hyperparameter τ 100.

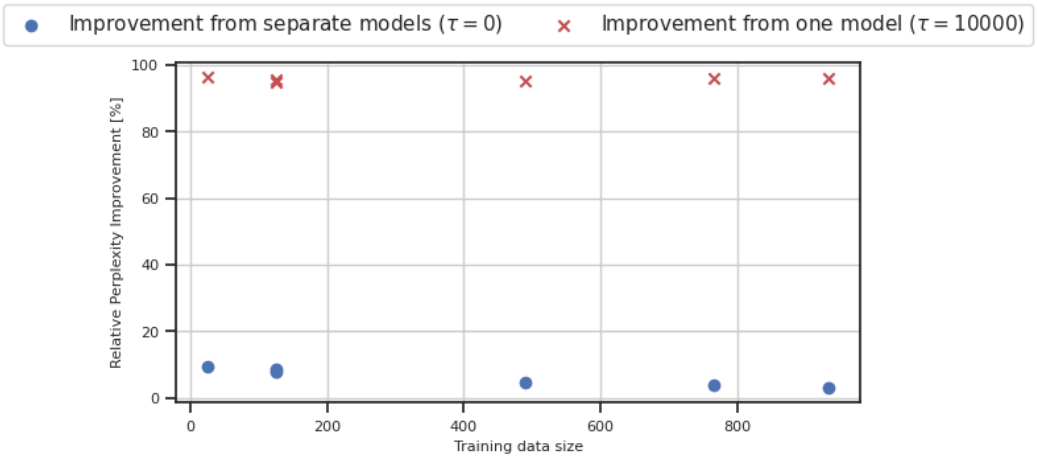

Figure 4: Relative improvement in validation data perplexity in comparing the one model for all tasks and one task per model approaches to the model with a learning rate of 0.001 and τ = 100.

Anger Log Likelihood	-3.948127
Fear Log Likelihood	-2.970546
Joy Log Likelihood	-3.206795
Sadness Log Likelihood	-3.046985
Surprise Log Likelihood	-3.232905
Love Log Likelihood	-3.072368
Average Log Likelihood	-3.280586
Perplexity	27.610291
Loss	7953.187988

Table 2: Final log-likelihoods for testing data tasks, overall log-likelihood, perplexity, and loss.

4.1 Conclusion

In the end, we were able to adapt the BoRA framework from Eide and Frigessi [2] to a new language model and dataset. Using `GPT2` and the Spotify lyrics dataset [1], the pre-trained language model was fine-tuned according to the new tasks and data and showed an improvement from training and validation to testing data set. The best hyperparameters for this language model and dataset were a learning rate of 0.001 and τ being 100, where τ is representative of how similar the task parameters are to one another, signifying the balance between learning all tasks as one model and learning each task as an individual model. This framework proves to be able to balance the two approaches and provides a useful methodology in fine-tuning large language models for task-specific use.

References

[1] DevDope. 500k+ spotify songs with lyrics,emotions & more, 2025. Hosted on Kaggle.

[2] Simen Eide and Arnoldo Frigessi. Bora: Bayesian hierarchical low-rank adaptation for multi-task large language models. In *Proceedings of the 6th Northern Lights Deep Learning Conference (NLDL)*, volume PMLR 265, 2025.

[3] Edward J. Hu, Yelong Shen, Phil Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shixiang Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2106.09685.

[4] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2018.

[5] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.