



Autosaved just  
now

🔒 Edits visible only to you

Publish to project



# Exploring Project Weights & Biases for Fine-Tuning a Large Language Model with LoRA

Add a description...

Po

## TLDR

- Learning that there are different techniques of LoRA fine tuning, such as LLaMA Factory, PEFT and keras, etc.
- We can add additional trainable layers to a PEFT adapter.
- There are more parameters of LoRA if we use PEFT technique, but not much in keras.nlp.
- LoRA parameters:

### rank:

Higher rank, more trainable parameters (more memory and computing cost). Usually set rank as 4, 8, 16, or 32.

### alpha (scaling factor):

If alpha is too large, the fine-tuning process may be unstable; if it is too small, it may not learn enough adjustments.

### trainable:

When set to True, the LoRA layers (A, B matrices) will be updated during training; set to False to freeze and not update.

Generally, it is set to True during fine-tuning, but sometimes fixed LoRA weights are tested first for inference and comparison.

- We've learned from the course about how to calculate the precision. In this project, we learn more about that the precision is not only affect the results, but also the GPU resources (times, power, temprature).

Overall, I'll recommand people about using W&B and take more class here!

## Abstract

Large Language Models (LLMs) have demonstrated state-of-the-art performance across various natural language processing tasks. However, fine-tuning these models often requires significant computational resources. Low-Rank Adaptation (LoRA) offers a parameter-efficient alternative, enabling effective fine-tuning by injecting trainable low-rank matrices into the model's architecture. This report details the process of fine-tuning a large language model (Gemma) using LoRA (Low-Rank Adaptation) and tracking experiments with Weights & Biases (W&B). We describe the model architecture, data preprocessing, training setup, and experimental outcomes. Our experiments, performed using KerasNLP on a subset of the Databricks Dolly-15k dataset, demonstrate how LoRA can significantly reduce the number of trainable parameters while maintaining competitive performance.

## Project Objective

This project aims to fine-tune the 2B-parameter GemmaCausalLM using LoRA (rank 4) on a subset of the Databricks Dolly 15k dataset. We integrate Weights & Biases (wandb) to track metrics and resource utilization.

## Environment and Tools

- Hardware:

GPU: NVIDIA A100 (40GB)

CPU Cores: 6–12 (varies by Colab environment)

RAM: ~90GB available

- Software:

Python: 3.11.11

Keras / KerasNLP: Keras >= 3, keras-nlp (latest version)

Weights & Biases: wandb 0.19.6

## Model

- Base Model: GemmaCausalLM (“gemma2\_2b\_en” preset)
- Parameter Count: ~2.6 billion parameters in full precision
- LoRA Integration:

Rank tested: 4

Mixed precision policy: FP32 (default), FP16, BF16

## Dataset

- Source: Databricks Dolly 15k

- Usage:

Filtered out examples with context for simplicity

Took a subset of 1000 samples to expedite experiments

- Format:

Combined instruction and response into a single string for causal LM training

#### Example template 1:

*Instruction:*

*What should I do on a trip to Europe?*

*Response:*

*{response}*

#### Example template 2:

*Instruction:*

*Explain the process of photosynthesis in a way that a child could understand.*

*Response:*

*{response}*

### **Weights & Biases Configuration**

- Project: gemma-lora-finetuning
- Project path: [ad2000x-none/LoRA\\_gemma\\_keras\\_test/7y41nv8o](#)
- Key Logged Metrics:

loss

sparse\_categorical\_accuracy

learning\_rate

- System Monitoring:

GPU Memory Allocated, GPU Utilization (%)

CPU Utilization, Disk I/O, Network Traffic

## Methodology and Technical Details

- Fine-Tuning Strategy with LoRA

The model is based on the GemmaCausalLM architecture using KerasNLP. Initially, the full model (approximately 2.6 billion parameters) is loaded and its summary is inspected. We employ LoRA with a rank of 4. The key idea is to inject low-rank updates into the backbone of the Gemma model so that only these updates are trainable during fine-tuning. This method reduces the trainable parameter count from over 9 GB to approximately 11 MB while the vast majority remain frozen.

- Integration with Weights & Biases

Weights & Biases (W&B) is integrated into the training loop to track experiments. Initialization and logging are done as follows:

```
import wandb

wandb.init(

    project="LoRA_gemma_kerasNLP_W&B",
    name="FP32",
    config={
        "epochs": 20,
        "batch_size": 4,
        "learning_rate": 5e-5,
        "weight_decay": 0.01,
        "lora_rank": 4
    }
)
```

The W&B integration tracks metrics such as loss and sparse categorical accuracy during training. Additional callbacks, including early stopping, help in monitoring and optimizing the training process.

- Training Workflow

Loading and preprocessing the dataset.

Compiling the model with an AdamW optimizer, excluding layer norm and bias from weight decay to ensure that their effects are not weakened by regularization, thereby improving the overall model performance and training stability.

Running the training loop with early stopping to ensure that training is efficient and captures the best model state and W&B logging.

Evaluating the model before and after fine-tuning with specific prompts to test generalization and response quality.

Pre-Fine-Tuning Inference

Before fine-tuning, the model was queried using prompts such as “What should I do on a trip to Europe?” and “Explain photosynthesis in a way that a child could understand.” The responses were generic and demonstrated limitations in adapting to specific instructions.

Post-Fine-Tuning Performance

After applying LoRA fine-tuning:

- Trip to Europe Prompt: The model provided detailed itineraries including city-specific recommendations.
- Photosynthesis Prompt: The model’s explanation became more structured and accessible, though some responses still required simplification for a younger audience.

Metric Trends

During training, loss consistently decreased from approximately 0.84 to 0.38 while sparse categorical accuracy improved from around 54% to over 76%. These trends were recorded in W&B, which provided real-time dashboards for metric visualization.

Trainable Parameters Comparison

```
1 gemma_lm = keras_nlp.models.GemmaCausalLM.from_preset("gemma2_2b_en")
2 gemma_lm.summary()
```

Preprocessor: "gemma\_causal\_lm\_preprocessor"

Layer (type)	Config
gemma_tokenizer (GemmaTokenizer)	Vocab size: 256,000

Model: "gemma\_causal\_lm"

Layer (type)	Output Shape	Param #	Connected to
padding_mask (InputLayer)	(None, None)	0	-
token_ids (InputLayer)	(None, None)	0	-
gemma_backbone (GemmaBackbone)	(None, None, 2304)	2,614,341,888	padding_mask[0][0], token_ids[0][0]
token_embedding (ReversibleEmbedding)	(None, None, 256000)	589,824,000	gemma_backbone[0][0]

Total params: 2,614,341,888 (9.74 GB)  
Trainable params: 2,614,341,888 (9.74 GB) ✓  
Non-trainable params: 0 (0.00 B)

Original parameters amount


```
1 # Enable LoRA for the model and set the LoRA rank to 4.
2 gemma_lm.backbone.enable_lora(rank=4)
3 gemma_lm.summary()
```

Preprocessor: "gemma\_causal\_lm\_preprocessor"

Layer (type)	Config
gemma_tokenizer (GemmaTokenizer)	Vocab size: 256,000

Model: "gemma\_causal\_lm"

Layer (type)	Output Shape	Param #	Connected to
padding_mask (InputLayer)	(None, None)	0	-
token_ids (InputLayer)	(None, None)	0	-
gemma_backbone (GemmaBackbone)	(None, None, 2304)	2,617,270,528	padding_mask[0][0], token_ids[0][0]
token_embedding (ReversibleEmbedding)	(None, None, 256000)	589,824,000	gemma_backbone[0][0]

Total params: 2,617,270,528 (9.75 GB)  
Trainable params: 2,928,640 (11.17 MB)   
Non-trainable params: 2,614,341,888 (9.74 GB)

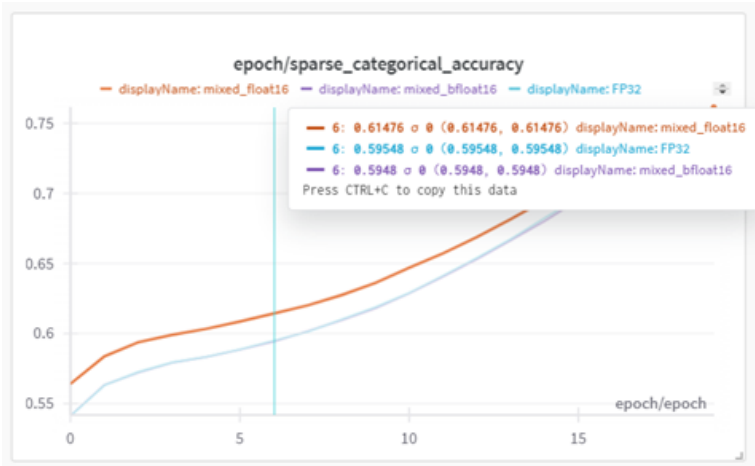
LoRA applied parameters amount

We can observe that trainable parameters decrease significantly from 2B to 2M.

Comparison Across Mixed Precision Modes

Experiments were also conducted with different precision settings (FP32, BF16, FP16). Although the core fine-tuning behavior remained similar, slight differences in training time and loss convergence were observed.

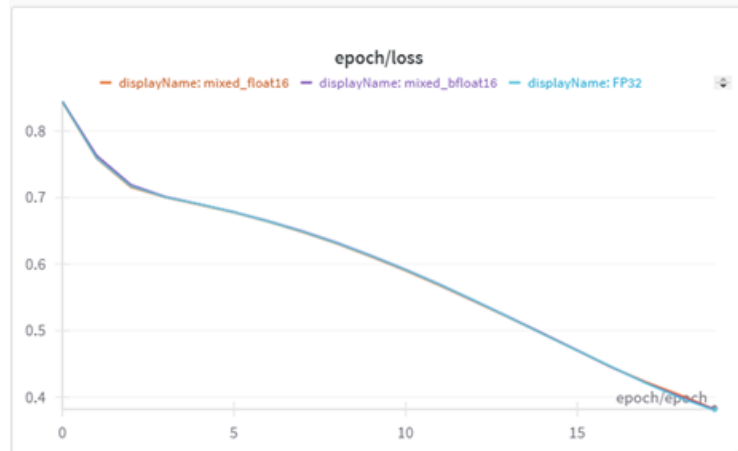
- Accuracy:



Higher accuracy under mixed float16NVIDIA A100 Tensor Cores with Tensor Float (TF32) provide up to 20X higher performance over the NVIDIA Volta with zero code changes and an additional 2X boost with automatic mixed precision and FP16.

<https://www.nvidia.com/en-us/data-center/a100/>

- Loss:



It's all about rounding of decimal places.

Not perfectly accurate.

FP16 noise actually provides a "random regularization" effect, similar to dropout or weight noise, which helps the model avoid overfitting. Each time the gradient is calculated and the model parameters are updated, the actual updated value will deviate slightly from the theoretical value due to rounding and precision loss. These deviations are equivalent to adding small random perturbations to the parameters.

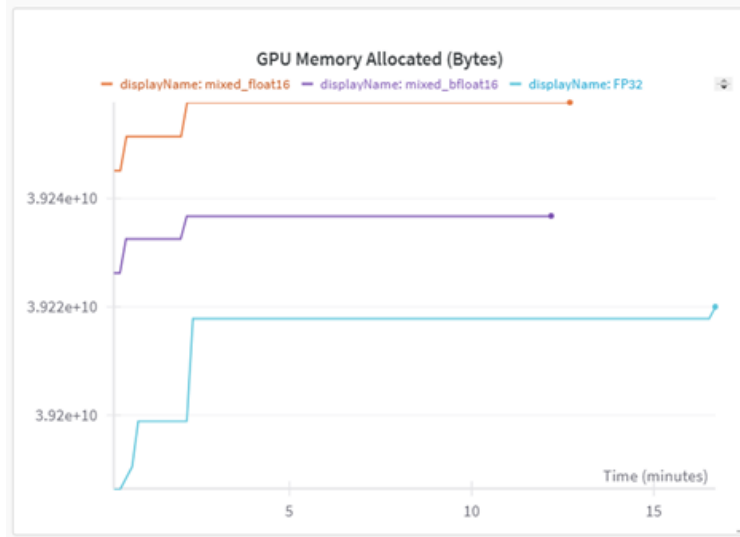
Apply a "Blur filter" to the gradient.

Lower precision may produce a smoother optimization surface, allowing the optimizer to escape from sharp local minima more easily. Low precision (such as FP16) will "blur" small changes that would otherwise be calculated with high precision because of rounding of decimal places.

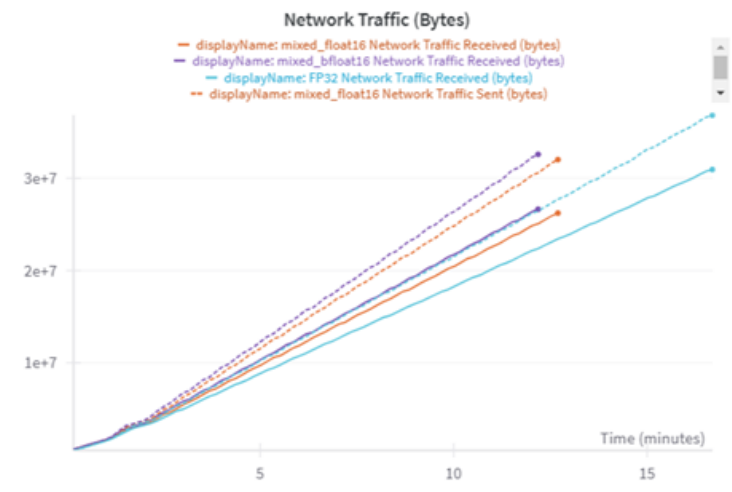
The A100 GPU's Tensor cores are optimized for FP16, which may provide more consistent computing behavior.

<https://docs.nvidia.com/deeplearning/tensorrt/latest/inference-library/accuracy-considerations.html>

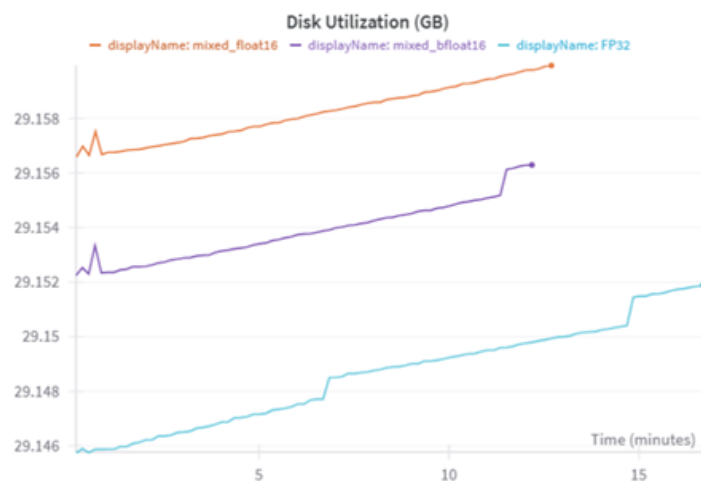
**GPU Memory Allocated (Bytes)**



LoRA maintains some FP32/low-rank weights, temporary blocks, etc. in FP16 mode, increasing the surface configuration amount.

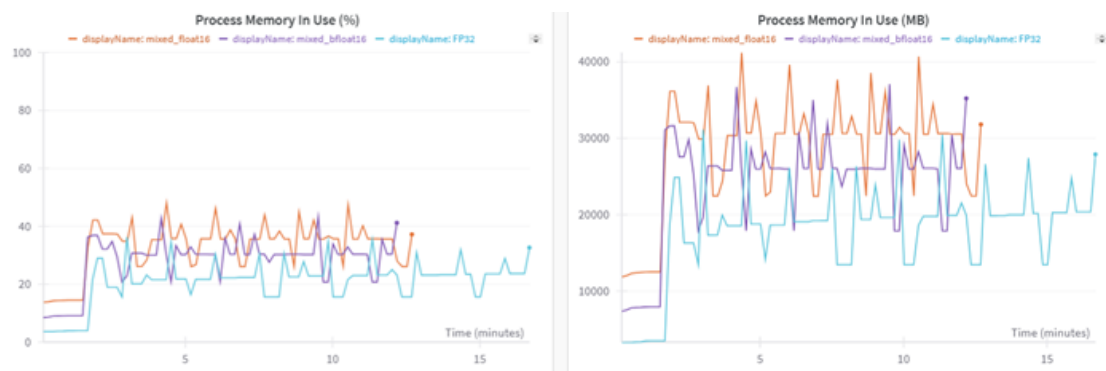


FP32 precision model allocates slightly less GPU memory, it generates more network traffic, likely because the 32-bit metadata needs to be transferred more frequently between different devices.





FB16 uses the most disk space, likely due to the need to store additional intermediate states or caches to handle mixed precision training.



FP16 mode often requires a copy of FP32 "master weights" to accumulate gradients, or requires additional buffers in the framework for precision conversion and gradient accumulation, so mixed precision uses about 10-20GB more memory than FP32



Mixed precision formats (FP16/BF16) consume more system memory resources, probably because they need to maintain buffers of multiple precision formats. That is, although mixed precision formats have advantages in GPU computing efficiency, they do use more memory resources to achieve this efficiency.

## Discussion

### Inference Texts Comparison

		Instruction	
	LoRA rank=4	What should I do on a trip to Europe?	Explain the process of photosynthesis in a way that a child could understand.
Inference before fine tuning	n/a	If you have any special needs, you should contact the embassy of the country that you are visiting. You should contact the embassy of the country that I will be visiting.	Plants need water, air, sunlight, and carbon dioxide. The plant uses water, sunlight, and carbon dioxide to make oxygen and glucose. The process is also known as photosynthesis.
Inference after fine tuning	FP32	When visiting a new place, it's best to create a rough itinerary that consists of the most important sites and things to do. For example, when in Paris, you should create a rough itinerary that includes the Eiffel Tower, Arc de Triomphe, Musee d'Orsay, Musee de Orsay, Musee de l'Anmilinium and Musee National du Chateau de Versailles. It's also recommended to do a lot of exploring on the streets, to find "chambres d'il y a plus" (hidden gems) that are off the beaten path.	Chlorophyll foundInstruction: Chlorophyll found in plants uses the sun's energy to make food for the plant. The plant puts most of its food making energy into the plant's roots and stems. The sunlight shining through the plant's leaves causes chlorophyll toInstruction: Sunlight shining through the leaves causes chlorophyll toInstruction: Causes chlorophyll toInstruction: Use chlorophyll toInstruction: Use chlorophyllInstruction: Absorbs the sun's energy and converts it into chemical energy using carbon dioxide and water. The plant releases the chemical energy in the form of plant Instruction: Releases the chemical energy in the form of plant Instruction: Plants Instruction: Plants Instruction: PlantsInstruction: PlantsInstruction:
	FP16	This is a great trip to Europe. You can visit many cities in the different countries. You can visit the museums, parks and architecture. You can also visit some historical places. You can also visit some famous landmarks like the Eiffel Tower in Paris, France and the Accademy in Rome, Italy. You can also take day trips to the surrounding areas.	Photosynthesis is the process by which plants, some bacteria and some other organisms use the sun's energy to convert carbon dioxide and water into glucose and oxygen. This process is also known as making "food" from the sun. The process occurs in a part of the cell called the chloroplast. Inside the chloroplast are tiny structures known as * * * (* *). * * * are responsible for capturing the sun's energy and * * * it. This captured * * * is then * * * by * * * to produce sugar and oxygen. The oxygen produced during this process is released as * * * into the atmosphere.
	BF16	Europe is one of the most beautiful continents in the world. There are a ton of cities to visit and a lot of different countries to explore. Here's some ideas of what you should do on a trip to Europe.  First, you should decide where you want to go and how long you want to stay. There are 27 countries in Europe so you might want to pick one of the most popular destinations like Paris, London, Berlin, Rome, Barcelona, or Prague. Once you decide on the country, you can look up the currency and exchange rate to that country, and how much money to money exchanges usually charge.  The next thing to do is make a reservation for your accommodations. Most people recommend booking a Airbnb or a hotel, as they are usually a little cheaper than booking sites like Booking.com. When booking an accommodation, you should pick one that's close to a metro or train station so you can easily get around the city. You should also pick something that has good reviews, to ensure you won't be disappointed.	The process of photosynthesis is how plants and some other organisms turn the energy of sunlight into sugar that they can use as food. Here's how it works: 1) Sunlight enters a plant through tiny holes called stomata, which are located on the underside of leaves. 2) Sunlight is absorbed by specialized cells called chloroplasts. These are found in the interior of plant cells. 3) Within the chloroplasts, sunlight is converted into chemical energy in the form of ATP (adenosine triphosphate). At the same time, chlorophyll molecules capture light in photosynthesis and water molecules (H2O) are oxidized ( "oxygen is removed"). This water is then split into oxygen and hydrogen (H2). Hydrogen is released as oxygen as the plant photosynthesizes. 4) Carbon dioxide (CO2) from the air is converted into sugar (glucose) with the use of ATP to form an unstable sugar called RuBP (ribose 1,5-bisphosphate). 5) As the process of photosynthesis continues, the unstable RuBP is "fixed" with electrons from the hydrogen (H2)

## Human Evaluation

		Instruction	
	LoRA rank=4	What should I do on a trip to Europe?	Explain the process of photosynthesis in a way that a child could understand.
Inference before fine tuning	n/a	Completeness: Doesn't properly answer the question.	Readability: It is easy to understand for an adult.
Inference after fine tuning	FP32	Completeness: Provides a general answer and mentions some specific tourist attractions, but lacks a detailed plan.	Readability: Jumps straight into the explanation of chlorophyll, with a poorly structured response.
	FP16	Completeness: Provides a general answer and mentions some specific tourist attractions, but lacks a detailed plan. Also mentions a time scale.	Readability: Generates some garbled text but uses metaphors to explain the process.
	BF16	Completeness: Provides a more specific plan, including locations, currency, and hotels. Also provides information about what users should do. The answer is cut off due to the length limitation.	Readability: A very specific answer with a process explanation, but it may not be easy for a child to understand.

- FP32 is more stable, but the batch size during fine-tuning may be smaller, resulting in limited model training results so it doesn't work as our wish.
- The weight updates of FP16 may be affected by the value range, resulting in insufficient learning and thus a more concise response.

- BF16 can better maintain the characteristics of FP32 training, but at the same time reduce memory usage and make the response more detailed.

Evaluation Metric

		Instruction			
		What should I do on a trip to Europe?		Explain the process of photosynthesis in a way that a child	
LoRA rank=4		Perplexity (Lower is Better)	Readability (Flesch Score, Higher is Easier)	Perplexity (Lower is Better)	Readability (Flesch Score, Higher is Easier)
Inference before fine tuning	n/a	1.3992	64.2	2.0194	53.58
Inference after fine tuning	FP32	2.9536	51.52	1.6566	28.17
	FP16	2.3916	61.12	3.1765	58.58
	BF16	2.8741	69.62	2.3551	46.27

We choose Perplexity and Readability as metrics and find that the reply quality varies depending on the question. For the trip-related question, the perplexity is the lowest before fine-tuning. However, for the photosynthesis question, FP32 achieves the best performance. Readability, BF16 performs best in trip question and FP16 performs best in photosynthesis question.

Conclusions

- The different precision significantly affect the fine-tuning results. In this experiment, the performance of BF16 performs best overall, considering the training resources and the inference. Overall, for the dataset, LoRA rank=4 and with BF16 fine-tuning is recommended for further runs.
- Weights and Biases are able to demonstrate the training situation in the backstage, and it is easy to integrate with our experiment.
- The evaluation methods and metrics can be tricky. In this project, we can find that the human feedback is quite different compare with the evaluation metric.

Limitation

- Only 1000 lines of dataset are selected for this experiment.
- Limits the length of generated answers.
- Only makes one configuration.

References

1. LoRA Paper: Low-Rank Adaptation of Large Language Models by Edward Hu et al. [<https://arxiv.org/pdf/2106.09685>]

2. Gemma Documentation: Google’s Gemma Models [<https://huggingface.co/google/gemma-2-2b>]

3. KerasNLP: KerasNLP Documentation [[https://keras.io/api/keras\\_nlp/](https://keras.io/api/keras_nlp/)]

4. Databricks Dolly 15k: Hugging Face Dataset [<https://huggingface.co/datasets/databricks/databricks-dolly-15k>]

5. Weights & Biases: W&B Documentation [<https://docs.wandb.ai/guides/>]