

Task-1

By

Shashank Gupta

From Language to Light: A LoRA Bootcamp for Text-to-Vision Learning

Phase 1: LoRA Basics – Text Fine-tuning

Objective

Implement and evaluate **Low-Rank Adaptation (LoRA)** on a text classification task, and compare its performance with traditional full fine-tuning.

Task

- Fine-tuned `distilbert-base-uncased` for **IMDb sentiment classification**
 - Compared full vs LoRA tuning in terms of accuracy, efficiency, and model size
-

What is LoRA?

LoRA (Low-Rank Adaptation) is a parameter-efficient fine-tuning (PEFT) method. It introduces small trainable adapter layers and keeps the rest of the model frozen.

- Trains only a fraction of total parameters
 - Reduces memory and compute cost
 - Speeds up training
 - Adapter weights are lightweight and shareable
-

Results Summary

Metric	Full Fine-Tuning	LoRA Fine-Tuning	Key Insight
Accuracy	87.5%	87.0%	Comparable performance
Training Time	~22.6 minutes (1359s)	~19 minutes (1143s)	~15% faster
Trainable Parameters	~67M (100%)	~740K (1.09%)	Huge reduction
Model Size	~256 MB	~3 MB (adapter only)	Tiny adapter

How to Run

```
pip install transformers==4.40.0 datasets==3.6.0 peft==0.10.0 accelerate torch wandb  
numpy  
gurukul-lora-env\Scripts\activate  
python LoRA_Text/train_distilbert_full_vs_lora.py
```

Outputs:

- ./results/
- ./logs/
- ./LoRA_Text/
- ./wandb/

[Comparison Chart of Accuracy & Training Time] :

```
--- Starting Full Fine-tuning Phase ---
Some weights of DistilBERTForSequenceClassification were not initialized from the model checkpoint at distilbert-base-uncased and are newly initialized: ['classifier.bias', 'classifier.weight', 'pre_classifier.bias', 'pre_classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
✓ DistilBERT model (for full fine-tuning) loaded
Instantiating TrainingArguments for full fine-tuning...
✓ TrainingArguments for full fine-tuning instantiated.
✖ Starting FULL fine-tuning...
 0% | 0/250 [00:00<?, ?it/s]
:LoRA_TextToVision\gurukul-lora-env\lib\site-packages\torch\utils\data\dataloader.py:665: UserWarning: 'pin_memory' argument is set as true but no accelerator is found, then device pinned memory won't be used.
  warnings.warn(warn_msg)
{'loss': 0.6207, 'grad_norm': 14.1897602088129883, 'learning_rate': 2e-05, 'epoch': 0.4}
{'loss': 0.3878, 'grad_norm': 2.75079154962617, 'learning_rate': 6.666666666666667e-06, 'epoch': 0.8}
{'eval_loss': 0.30378642678260803, 'eval_accuracy': 0.875, 'eval_runtime': 128.5255, 'eval_samples_per_second': 7.781, 'eval_steps_per_second': 0.973, 'epoch': 1.0}
{'train_runtime': 1359.2685, 'train_samples_per_second': 1.471, 'train_steps_per_second': 0.184, 'train_loss': 0.45576555633544924, 'epoch': 1.0}
100% | 250/250 [22:39<00:00, 5.44s/it]
Full fine-tuning time: 1359.41 seconds
***** train_full metrics *****
epoch = 1.0
total_flos = 2467309F
train_loss = 0.4558
train_runtime = 0:22:39.26
train_samples_per_second = 1.471
train_steps_per_second = 0.184
C:\LoRA_TextToVision\gurukul-lora-env\lib\site-packages\torch\utils\data\dataloader.py:665: UserWarning: 'pin_memory' argument is set as true but no accelerator is found, then device pinned memory won't be used.
  warnings.warn(warn_msg)
100% | 125/125 [02:02<00:00, 1.02it/s]
Full fine-tuning evaluation metrics: {'eval_loss': 0.30378642678260803, 'eval_accuracy': 0.875, 'eval_runtime': 123.6929, 'eval_samples_per_second': 8.085, 'eval_steps_per_second': 1.011, 'epoch': 1.0}
***** eval_full metrics *****
epoch = 1.0
eval_accuracy = 0.875
eval_loss = 0.3038
eval_runtime = 0:02:03.69
eval_samples_per_second = 8.085
eval_steps_per_second = 1.011
Full fine-tuned model saved to ./LoRA_Text/distilbert_full_finetuned

--- Starting LoRA Fine-tuning Phase ---
C:\LoRA_TextToVision\gurukul-lora-env\lib\site-packages\huggingface_hub\file_download.py:943: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. If you want to force a new download, use 'force_download=True'.
  warnings.warn(
Some weights of DistilBERTForSequenceClassification were not initialized from the model checkpoint at distilbert-base-uncased and are newly initialized: ['classifier.bias', 'classifier.weight', 'pre_classifier.bias', 'pre_classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
✓ DistilBERT model with LoRA adapter initialized
trainable params: 739,586 || all params: 67,694,596 || trainableK: 1.0925332946813067
Instantiating TrainingArguments for LoRA fine-tuning...
✓ TrainingArguments for LoRA fine-tuning instantiated.
✖ Starting LoRA fine-tuning...
 0% | 0/250 [00:00<?, ?it/s]
{'loss': 0.5828, 'grad_norm': 2.16012883186340833, 'learning_rate': 0.0003, 'epoch': 0.4}
{'loss': 0.3992, 'grad_norm': 3.435936689376831, 'learning_rate': 9.999999999999999e-05, 'epoch': 0.8}
{'eval_loss': 0.30232303741607666, 'eval_accuracy': 0.87, 'eval_runtime': 122.5799, 'eval_samples_per_second': 8.158, 'eval_steps_per_second': 1.02, 'epoch': 1.0}
{'train_runtime': 1143.7404, 'train_samples_per_second': 1.749, 'train_steps_per_second': 0.219, 'train_loss': 0.45148636627197264, 'epoch': 1.0}
100% | 250/250 [19:03<00:00, 4.57s/it]
LoRA fine-tuning time: 1143.99 seconds
***** train_lora metrics *****
epoch = 1.0
total_flos = 2509716F
train_loss = 0.4515
train_runtime = 0:19:03.74
train_samples_per_second = 1.749
train_steps_per_second = 0.219
C:\LoRA_TextToVision\gurukul-lora-env\lib\site-packages\torch\utils\data\dataloader.py:665: UserWarning: 'pin_memory' argument is set as true but no accelerator is found, then device pinned memory won't be used.
  warnings.warn(warn_msg)
100% | 125/125 [02:02<00:00, 1.02it/s]
Full fine-tuning evaluation metrics: {'eval_loss': 0.30232393741607666, 'eval_accuracy': 0.87, 'eval_runtime': 123.8512, 'eval_samples_per_second': 8.074, 'eval_steps_per_second': 1.009, 'epoch': 1.0}
***** eval_lora metrics *****
epoch = 1.0
eval_accuracy = 0.87
eval_loss = 0.3023
eval_runtime = 0:02:03.85
eval_samples_per_second = 8.074
eval_steps_per_second = 1.009
C:\LoRA_TextToVision\gurukul-lora-env\lib\site-packages\huggingface_hub\file_download.py:943: FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0. Downloads always resume when possible. If you want to force a new download, use 'force_download=True'.
  warnings.warn(
LoRA adapter saved to ./LoRA_Text/distilbert_lora_adapter

--- Comparison Summary ---
Full fine-tuning time: 1359.41s, Accuracy: 0.875, Train Loss: 0.45576555633544924
LoRA fine-tuning time: 1143.90s, Accuracy: 0.87, Train Loss: 0.45148636627197264
wandb: Run summary:
wandb:   eval/accuracy 0.87
wandb:   eval/loss 0.30232
wandb:   eval/runtime 123.8512
wandb:   eval/samples_per_second 8.074
wandb:   eval/steps_per_second 1.009
wandb:   full_eval_accuracy 0.875
wandb:   full_final_accuracy 0.875
wandb:   full_total_time_seconds 1359.41247
wandb:   full_train_loss 0.45577
wandb:   full_tuning_time_seconds 1359.41247
wandb:   lora_eval_accuracy 0.87
wandb:   lora_final_accuracy 0.87
wandb:   lora_total_time_seconds 1143.90235
wandb:   lora_train_loss 0.45149
wandb:   lora_tuning_time_seconds 1143.90235
wandb:   total_flos 269478813696000.0
wandb:   train/epoch 1
wandb:   train/global_step 250
wandb:   train/grad_norm 3.43594
wandb:   train/learning_rate 0.0001
wandb:   train/loss 0.3992
wandb:   train/train_loss 0.45149
wandb:   train/runtime 1143.7404
wandb:   train_samples_per_second 1.749
wandb:   train_steps_per_second 0.219
wandb: You can sync this run to the cloud by running:
wandb: wandb sync C:\LoRA_TextToVision\wandb\offline-run-20250606_103421-d9aak201
wandb: Find logs at: \wandb\offline-run-20250606_103421-d9aak201\logs
✓ wandb session closed.
✖ Training script completed successfully!
○ (gurukul-lora-env) PS C:\LoRA_TextToVision>
```

Phase 2: LoRA for Vision – Image Fine-tuning with Stable Diffusion

Objective

Use LoRA to fine-tune **Stable Diffusion v1.4** and generate stylized Gurukul-themed images.

Task

- Trained LoRA adapter on **10 themed images**
 - Used adapter to generate images from prompts
 - Focused on reducing parameter update size and memory use
-

Technical Insights

Metric	Value	Insight
Trainable Parameters	297,984 (~0.035%)	Only adapters updated
Total Parameters	~859M	Core model untouched
Training Steps	10	Very low resource need
Adapter Size	~3 MB	Lightweight, reusable

Sample Prompts

- “AI robot teaching students in a village school”
 - “A traditional Gurukul reimagined with VR and AR”
 - “Students learning under a hologram tree”
-

[Generated Gurukul images]



1.A child meditating in a virtual garden



2. AI robot teaching students in a village school



3. Students learning under a hologram tree

💡 How to Run

```
pip install diffusers transformers peft accelerate xformers
gurukul-lora-env\Scripts\activate
python LoRA_StableDiffusion/train_lora_sd.py
python LoRA_StableDiffusion/inference_generate.py
```

📁 Outputs:

- ./LoRA_StableDiffusion/lora_output/
 - ./LoRA_StableDiffusion/outputs/
 - ./training_data/gurukul_hologram_style/
-

Phase 3: Visual Sequencing – Image-to-Video Integration

Objective

Convert sequential prompts into images and merge them into animated video clips.

Key Features

- Paragraph prompts are split and converted to images
 - Scenes saved in separate folders: `frames_scene_X`
 - Final videos saved in `video_outputs/`
-

Tools Used

- Hugging Face Diffusers + Stable Diffusion
 - Prompt Engineering
 - Python automation (`os, pathlib`)
 - **MoviePy** for frame stitching
-

 [Insert Image Placeholder – Side-by-side frames from one story scene]

 [Insert Image Placeholder – Screenshot of final video output folder]

Project Structure Overview

Folder	Description
<code>LoRA_Text/</code>	Text classification with LoRA
<code>LoRA_StableDiffusion/</code>	SD LoRA training and inference scripts
<code>VideoMaker/</code>	Code for video creation from image sequences
<code>lora_output/</code>	Saved adapter models
<code>outputs/</code>	Generated images
<code>training_data/</code>	Custom Gurukul-themed dataset used for Phase 2

Challenges Faced

- **⚠ Limited GPU VRAM**
 - **⚠ xFormers / triton unavailable** during training
 - **⚠ Maintaining object consistency across frames** is still tough
-

Future Scope

- Integrate ControlNet or img2img for smoother temporal transitions
 - Use Streamlit or Gradio to build a Web UI
 - Add scene tracking and frame-by-frame evolution
 - Explore AnimateDiff for dynamic video generation
-

Installation Steps

```
pip install -r requirements.txt  
gurukul-lora-env\Scripts\activate
```
