# DSA4213 Assignment 3

## Introduction

Pretrained models like BERT have become the integral in NLP applications. However, fully fine tuning all parameters of such models is computationally expensive and memory intensive. To address this, parameter fine tuning (PEFT) methods like LoRA (low rank adaptation) enable practitioners to adapt large models using a small number of additional trainable parameters.

Phishing is common in cybercrime where malicious URLs mimic legitimate sites to steal data. Traditional detection which is rule based and URL pattern matching fails against modern phishing attempts that use linguistic cues. Pretrained transformer models like BERT can learn semantic and contextual representations of URLs. We will fine tune the BERT model to the cybersecurity domain to classify phishing URLs.

## Dataset

We will use the Phishing Site Classification dataset (Shawin, 2023), hosted on Hugging Face. It has 3000 samples, labelled either phishing (0) or legitimate (1). The dataset is balanced between both classes, ensuring reliable evaluation metrics.

Since URLs are typically short, max sequence length was set to 64 tokens. The dataset was split into 80-10-10 train-validation-test with stratification to preserve label ratios.

## Model and Fine-tuning Strategies Used

We will compare performance of BERT with these fine tuning strategies:

1. Full fine tuning
   In full fine tuning, we update all parameters of BERT, including Transformer encoder layers.

2. LoRA (with ranks 4, 8 and 16)
   LoRa introduces a small trainable matrix into each attention layer while freezing the original pretrained weight. LoRA decomposes the large weight updates into products of 2 low rank matrices, thus drastically reducing the number of training parameters. We will experiment with ranks 4, 8 and 16.

## Experimental Setup

These are the hyperparameters we will use for all models:

- Seed = 42
- Optimizer: AdamW (weight decay = 0.01)
- Epoch = 3
- Batch size = 16
- Evaluation Strategy : epoch level
- Max sequence length = 64

These are the hyperparameters for LoRA (rank 4, 8, 16):

- Alpha = 16
- Dropout = 0.05

- Target Modules: Query and Value (since they give best overall performance)

We use a smaller learning rate for Full Finetuning to prevent large gradient updates from disrupting BERT's pretrained weights since all parameters are being optimised. LoRA fine-tuning is only on a few low rank adapter matrices while keeping most parameters frozen hence a higher learning rate was used to accelerate coverage of these light weight layers.

**Results**

These were the runtimes of all the models on training dataset:

| Models | Train Runtime |
|---|---|
| Full finetune | 97.8072 |
| LoRA rank 4 | 55.318 |
| LoRA rank 8 | 51.0366 |
| LoRA rank 16 | 51.6979 |

LoRA (around 55s) consistently took less runtime than full finetune (around 98s), showing that LoRA is computationally faster.

As the rank of LoRA increased, the runtimes increased, making LoRA slower.

| | model | accuracy | f1 | precision | recall | roc_auc |
|---|---|---|---|---|---|---|
| 0 | full fine-tune | 0.9156 | 0.9155 | 0.9177 | 0.9162 | 0.9781 |
| 1 | lora r=4 | 0.8556 | 0.8555 | 0.8562 | 0.8559 | 0.9223 |
| 2 | lora r=8 | 0.8533 | 0.8533 | 0.8544 | 0.8538 | 0.9226 |
| 3 | lora r=16 | 0.8511 | 0.8511 | 0.8520 | 0.8516 | 0.9239 |

*Figure 1 Accuracy, F1 score, Precision, Recall and ROC-AUC of all models*

Across all metrics, full fine tuning outperformed the LoRA variant. While full fine tuning achieved the best metrics, LoRA was comparable in performance to full fine tuning, differing by only 5-7% across all metrics.

When the rank of LoRA increased, performance of LoRA decreased across all metrics.

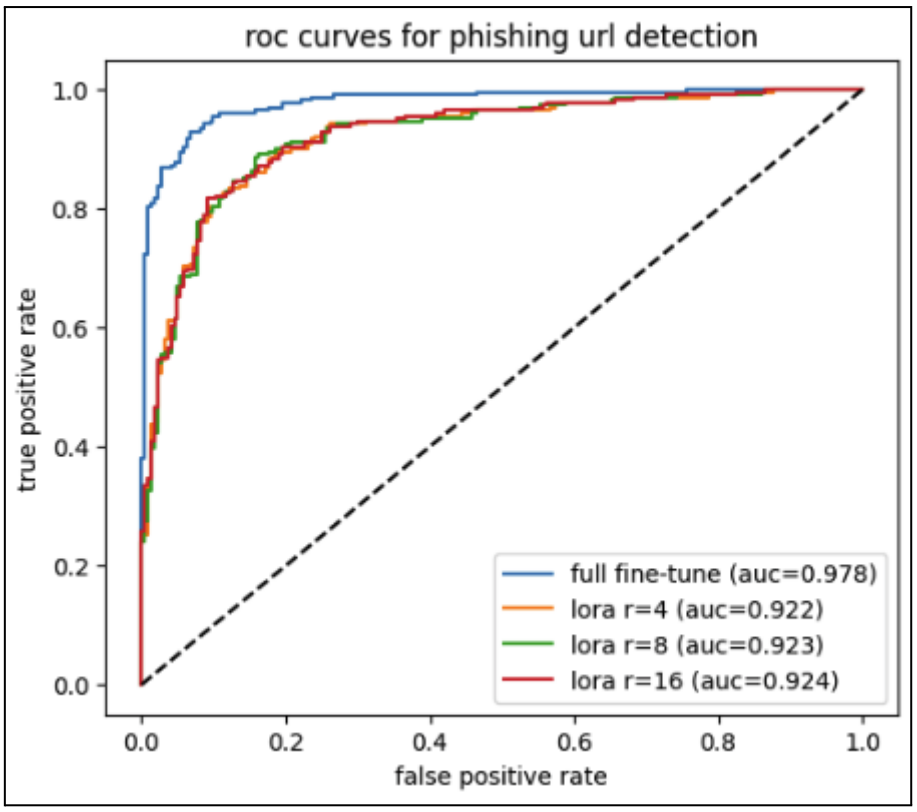


*Figure 2 AUC-ROC curve of all models*

The full fine tuned model achieved the highest AUC (0.978) showing strong separability between phishing and legitimate examples. The LoRA models show closely overlapping ROC curves with AUC values between 0.922-0.924, differing from full fine tuning by only 5%, indicating they are comparable with full fine tuning despite fewer parameters.

As the rank of LoRA increased, the AUC score increased slightly, indicating that as rank of LoRA increased, LoRA performs more similarly to full fine tuning.

**Discussion and Analysis**

Even though LoRA's performance lagged behind from full finetuning, its efficiency benefits are substantial. Training parameters are reduced by >90%, training time reduced by ~40-50%, whilst having comparable ROC-AUC (0.92 vs 0.98) indicating LoRA remains viable for fast low resource adaptation.

The ablation study of LoRA ranks indicates there is minimal sensitivity to rank hyperparameters over this dataset, with rank 4 already achieving most of the attainable performance. This shows that increase in rank does not linearly translate to performance gains especially for small scale fine-tuning tasks.

---

**References and Citations**

1. *Shawhin/Phishing-Site-Classification · Datasets at Hugging Face*. 1 Sep. 2024, https://huggingface.co/datasets/shawhin/phishing-site-classification.

2. chun, rachel. *Chxlz/DSA4213-Assignment-3*. 17 Oct. 2025, Python. 17 Oct. 2025. *GitHub*, https://github.com/Chxlz/DSA4213-Assignment-3.