CS190I: Generative AI, Spring 2025
Programming Assignment 2

Joe Lee

May 10, 2025

1 Model and Dataset Selection

I implemented fine-tuning experiments using the T5-small model [1] for headline generation. The model was fine-tuned on the Gigaword 10k dataset [2], which consists of news articles and their corresponding headlines. The task involves generating concise headlines from full news articles, making it an ideal candidate for testing different fine-tuning approaches.

2 Implementation Details

2.1 Fine-tuning Approaches

I implemented three distinct fine-tuning strategies:

1. Full fine-tuning: Updates all model parameters
2. Adapter fine-tuning: Adds trainable adapter layers
3. LoRA fine-tuning: Implements low-rank updates with varying ranks

Method	BLEU	ROUGE-1	ROUGE-2	ROUGE-L
Base Model	4.85	0.30	0.10	0.27
Full Fine-tuning	18.75	0.44	0.22	0.42
Adapter	5.02	0.30	0.10	0.27
LoRA (r=4)	13.43	0.39	0.17	0.37
LoRA (r=8)	12.97	0.38	0.17	0.36
LoRA (r=16)	12.48	0.37	0.16	0.35

TABLE I
PERFORMANCE METRICS ACROSS DIFFERENT FINE-TUNING METHODS.

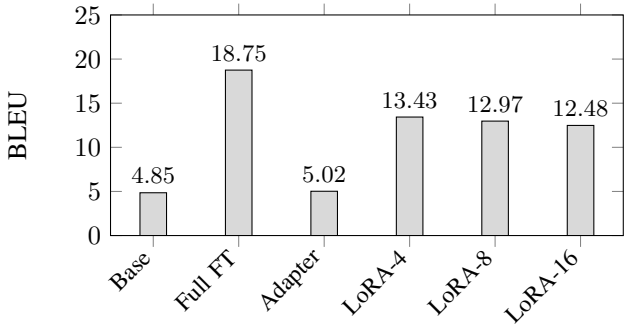

Fig. 1. BLEU scores across different fine-tuning methods.

2.2 Training Configuration

Training was performed on an NVIDIA RTX 3090 24GB GPU with the following parameters:

- Batch size: 32
- Learning rate: 3e-4
- Optimizer: AdamW
- Evaluation metrics: BLEU, ROUGE-1, ROUGE-2, ROUGE-L

Method	Time (s)
Full Fine-tuning	549
Adapter	432
LoRA (r=4)	426
LoRA (r=8)	427
LoRA (r=16)	462

3 Program Usage

The implementation includes three main components:

- `train.py`: Training script with configurable parameters for model type, output directory, batch size, epochs, and learning rate
- `eval.py`: Evaluation script for computing metrics and saving results
- `inference.py`: Script for generating headlines from input text using the fine-tuned models

Please check `README.md` for more details on how to run the program.

4 Analysis and Discussion

4.1 Accuracy Improvements

The results demonstrate significant improvements over the base model:

- Full fine-tuning achieved the best performance (BLEU: 18.75)
- LoRA fine-tuning provided a good balance between performance and efficiency
- Adapter fine-tuning showed minimal improvement over the base model

4.2 Impact of LoRA Rank

Analysis of different LoRA ranks reveals:

- Lower ranks (r=4) achieved better performance
- Higher ranks led to slightly worse results
- Training time increased with rank size

5 Lessons Learned

- Full fine-tuning provides the best accuracy but requires more computational resources
- LoRA offers an excellent trade-off between performance and efficiency
- Adapter fine-tuning may require more careful architecture design for this specific task
- Lower LoRA ranks can be more effective than higher ones, suggesting that the task doesn't require complex parameter updates

References

[1] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1-67, 2020.

[2] A. Mafzal, "Gigaword 10k Finetuning Dataset," Hugging Face Datasets, 2023. [Online]. Available: https://huggingface.co/datasets/anumafzal94/gigaword_10k_finetuning