From BART to Edge: Exploring Optimal Strategies for High-Quality Computing with Limited Resources

Xuanyou Liu, Prekshi Vyas, Raksha Ramesh, Manurag Khullar

December 2024

Abstract

Machine Translation (MT) is a pivotal task in Natural Language Processing (NLP), offering the potential to bridge linguistic barriers and enhance global communication. This project explores the challenges of deploying high-quality Chinese-to-English MT models on resource-constrained devices. By leveraging the UM-Corpus dataset, techniques such as Low-Rank Adaptation (LoRA) and layer freezing were investigated to optimize model performance while minimizing computational overhead. The findings demonstrate that parameter-efficient fine-tuning approaches can balance performance and resource constraints, making advanced MT capabilities feasible on edge devices. This work provides a comprehensive evaluation of MT strategies, offering insights into achieving scalable and efficient translation models for real-world applications.

1 Introduction

MT seeks to automatically convert text from one language into another, enabling seamless communication across linguistic borders. By providing access to content in languages users do not speak, MT supports international collaboration, global business endeavors, and broadens the accessibility of educational and cultural materials. As research advances, a key concern emerges: how can we maintain strong translation performance while operating under the computational constraints of edge devices such as smartphones, tablets, or specialized embedded systems?

This project zeroes in on translating from Chinese (Zh) to English (En) using the UM-Corpus dataset. The language pair offers a rich testing ground due to the distinctive orthography and syntactic structures of Chinese compared to English, challenging the model to capture not only lexical equivalences but also nuanced syntactic and semantic transformations. While large-scale MT models have shown exceptional quality, their deployment demands substantial computational and memory resources. This often makes them impractical for on-device applications. In contrast, smaller models, though easier to deploy on such devices, frequently struggle to match the performance of their larger counterparts.

Chinese Input: "我喜欢这本书。"

English Output: "I like this book."

Figure 1: An illustration of the core MT task.

Formal Definition of the Problem: Let $\mathcal{X}$ be the set of source language sentences in Chinese and $\mathcal{Y}$ be the set of target language sentences in English. We aim to learn a function $f : \mathcal{X} \to \mathcal{Y}$ that, for each $x \in \mathcal{X}$, produces a $y \in \mathcal{Y}$ which accurately reflects the meaning of x, respects target language grammar, and achieves high quality as measured by the chosen metric (BLEU Score). The function f operates within the bounds of available hardware resources, such as memory and processing power, suitable for deployment on edge devices or cloud servers.

Our primary motivation for this project is to discover effective methods to build and adapt MT models for resource-limited environments. We want to explore approaches that enable strong translation performance without significant computational overhead, making it feasible to train and deploy the models on devices with limited hardware capabilities. Techniques like LoRA (Low-Rank Adaptation) and selective parameter updating (e.g., layer freezing) allow us to fine-tune small models (e.g., mBART, M2M100) efficiently, potentially closing the gap in performance between resource-heavy large models and lightweight models suitable for on-device use. By doing so, we aspire to bring advanced MT capabilities to resource-constrained platforms, ultimately broadening the availability of high-quality, real-time translation services across a range of practical settings.

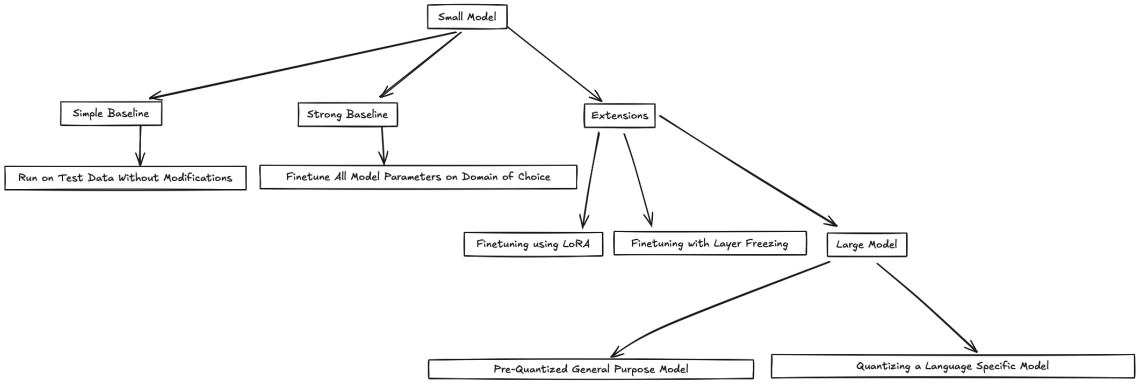

Figure 2: An illustration of the Experiments Conducted.

2 Literature Review

2.1 Datasets and Out-of-Domain Evaluation

A critical starting point for building robust Neural Machine Translation (NMT) models is the availability of high-quality parallel corpora. The UM-Corpus [6] serves as a foundational English-Chinese dataset originally constructed for Statistical Machine Translation (SMT), comprising approximately 15 million parallel sentences across eight diverse domains (e.g., news, laws, subtitles, and microblogs). Such domain breadth is pivotal for training NMT systems that can handle varied linguistic phenomena and maintain strong performance across multiple content types. Leveraging the UM-Corpus in this project thus aims to ensure that models remain resilient under realistic deployment conditions, especially on resource-constrained edge devices.

However, relying solely on in-domain test sets can misrepresent true generalization capabilities. To address this, recent research proposes more nuanced out-of-domain (OOD) challenge sets. Chen et al. [2] introduced a multifaceted evaluation framework for Chinese-English translation, categorizing sentences by word difficulty, length, grammatical complexity, and model learning difficulty. Their curated challenge sets, comprising 2,000 sentences per direction evenly divided into four difficulty tiers, offer a rigorous basis for testing how models perform under linguistic and structural stressors. Incorporating such OOD evaluations helps ensure that fine-tuned models remain robust under diverse and unpredictable input conditions, a crucial requirement for on-device MT applications.

2.2 Evaluation Metrics and Multi-faceted Quality Assessment

Selecting appropriate evaluation metrics is central to understanding model performance holistically. Although surface-form metrics like BLEU [4] have long been standards, they may fail to capture subtle semantic nuances. Embedding-based metrics such as BERTScore [7] offer deeper semantic sensitivity, but exhibit weaknesses in recognizing synonyms. Chen et al. [1] examined a broad range of 28 metrics, highlighting the need for a balanced approach: embedding-based metrics excel at detecting semantic anomalies, while traditional measures like BLEU and chrF [5] provide stable, domain-agnostic performance benchmarks. Guided by these insights, this project employs a combination of metrics—potentially including BLEU, BERTScore, COMET, and BLEURT—to yield a more comprehensive quality assessment.

2.3 Parameter Efficient Fine-Tuning Techniques

Beyond dataset quality and evaluation rigor, computational efficiency is paramount for MT systems running on edge devices. To address resource constraints, Hu et al. [3] introduced Low-Rank Adaptation (LoRA), a parameter-efficient fine-tuning method that updates only a small subset of model parameters via rank-decomposition matrices. This approach achieves performance on par with full fine-tuning but at substantially reduced computational and memory costs. Integrating LoRA with advanced models like mBART and Flan-T5 supports the development of high-quality, lightweight MT solutions suitable for deployment in low-resource environments. By incorporating LoRA and similar techniques, the proposed workflow aims to deliver robust, scalable,

and efficient translation models.

In summary, these literature-driven insights guide the proposed methodology: using the domain-rich UM-Corpus, adopting a balanced metric (BLEU Score) for comprehensive assessment, and employing parameter-efficient fine-tuning methods like LoRA and Layer-freezing to ensure that the final models meet both performance and resource constraints.

3 Experimental Design

3.1 Data

For our experiments, we utilize a subset of the UM-Corpus (Zh-En) [6] which is a widely used parallel corpus for Chinese-English machine translation. The data consists of paired sentences, one in Chinese (Zh) and its corresponding translation in English (En). This resource is known for its diverse content across multiple domains, making it suitable for assessing domain-specific translation performance.

Split	**Samples**	**Avg. Tokens (En)**	**Avg. Tokens (Zh)**
Training	80,000	12.4	13.1
Development	10,000	12.5	13.0
Test	10,000	12.3	13.2

Table 1: Data Split and Statistics: Each sample corresponds to one English-Chinese sentence pair.

The data is slightly skewed towards general-domain translations. For our evaluation, we focus on two distinct domains: Science and Education. Within these domains, we note a distribution skew: Science-related sentences make up approximately 40% of the test set, while Education-related sentences account for the remaining 60%. Table 2 shows an example from the dataset.

English	Researchers observed a significant increase in gene expression under these conditions.
Chinese (Zh)	在这些条件下，研究人员观察到了基因表达的显著增加

Table 2: Example Sentence Pair from the Science Domain

3.2 Data Preparation and Preprocessing

3.2.1 Data Loading and Extraction

The dataset was obtained from a compressed `.zip` archive containing bilingual `.txt` files. Each file consisted of alternating English and Chinese lines, and every pair of lines constituted a unique translation unit.

3.2.2 Data Preprocessing and Sampling

For the baseline evaluation, a sample of 100,000 translation units was used. A smaller subset of 10,000 units was created for hyperparameter tuning experiments. The combined dataset was then shuffled using a fixed seed (42) to ensure reproducibility. Subsequently, the data was partitioned into training (80%), development (10%), and test (10%) sets.

3.2.3 Tokenization

All sentences were tokenized using a suitable tokenizer. After tokenization, sentences were either padded or truncated to a maximum length of 50 tokens. To exclude padding tokens from loss computation, they were replaced with `-100` in the target sequences. Finally, the processed data was converted into PyTorch-compatible datasets, providing `input_ids`, `attention_mask`, and `labels` for subsequent model training.

3.3 Evaluation Metric

We employ the BLEU [4] score as our primary evaluation metric. BLEU (Bilingual Evaluation Understudy) measures the n-gram overlap between machine-generated translations and reference translations. It is calculated as follows:

$$\text{BLEU} = \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \times \min\left(1, \frac{\text{output length}}{\text{reference length}}\right)$$

where p_n is the precision of n-grams, w_n are the weights (usually uniform), and the brevity penalty ensures that shorter translations are not unfairly favored. BLEU has been widely used in MT research, providing a standardized and reproducible way to compare different approaches.

3.4 Simple Baseline Performance

To begin, we evaluated the pre-trained models mBART and M2M100 without any domain-specific fine-tuning. Table 3 presents the baseline BLEU scores.

Domain	mBART Baseline BLEU	M2M100 Baseline BLEU
Science	0.1223	0.0231
Education	0.1169	0.0347

Table 3: Simple Baseline BLEU Scores without Finetuning.

These low scores highlight the difficulty of the task, especially for M2M100 in the Science domain. This baseline serves as a clear starting reference point, allowing us to measure subsequent improvement once domain-specific fine-tuning methods are applied.

4 Experimental Results

4.1 Establishing a Strong Baseline

To create a robust baseline for our translation task, we fine-tuned both our models, using the paired English–Chinese dataset. These models employ dedicated tokenizers (`MBart50Tokenizer` and `M2M100Tokenizer`) that were configured to facilitate bilingual fine-tuning. Prior to large-scale fine-tuning, a preliminary hyperparameter tuning stage was conducted on a smaller subset of the data to identify optimal training parameters.

Hyperparameter Tuning (10,000 Samples)

In the tuning phase, we systematically varied key hyperparameters to identify a configuration yielding stable training and improved validation performance.

Hyperparameter	Values Explored
Learning Rates	`[2e-5, 3e-5, 5e-5]`
Batch Sizes	`[8, 16]`
Number of Epochs	`[1, 2, 3]`
Evaluation Criteria	Training and validation losses

Table 4: Configurations explored during hyperparameter tuning.

Here are the best performing hyper-parameters:

Domain	Model	LR	Batch	Best Epoch	Loss	Runtime (s)	Samples/sec
Education	mBART	2e-05	16	1	1.915	4.48	223.22
Education	M2M100	2e-05	16	3	2.656	2.25	444.42
Science	mBART	2e-05	8	1	1.815	5.15	194.11
Science	M2M100	2e-05	16	1	2.086	2.32	431.81

Table 5: Performance comparison of models across domains.

Fine-Tuning with Selected Hyper parameters (50,000 Samples)

After identifying the optimal set of hyperparameters, we fine-tuned both mBART-50 and M2M100 models on the full 50,000-sample dataset. This step aimed to establish a high-quality baseline against which future improvements could be measured.

Model	Domain	Baseline	Fine-Tuned	Improvement (%)	Training Time (hours)
mBART	Science	0.1209	0.1997	63.28	1.46
mBART	Education	0.1169	0.1192	4.85	1.44
M2M100	Science	0.0222	0.1200	419.48	1.94
M2M100	Education	0.0342	0.0809	143.56	0.83

Table 6: Comparison of Baseline and Fine-Tuned Models Across Models and Domains

4.2 Extensions

4.2.1 Incorporating LoRA

i. **Hyperparameter Tuning**
We employed a hybrid approach by initially performing hyperparameter tuning on the base models to identify effective configurations (e.g., learning rate, batch size). After establishing these foundational settings, we applied LoRA to the models. Subsequently, we conducted lightweight hyperparameter tuning focused on LoRA-specific parameters, such as the rank value r, scaling factor α, and dropout rate, to further refine performance.

Below are the best parameter settings for hyperparameter tuning with LoRA:

Model	Domain	r	α	Dropout	Validation Loss
mBART	Science	8	64	0.0	2.024966
M2M100	Education	8	64	0.0	2.791431
M2M100	Science	8	64	0.0	2.874502
mBART	Education	8	64	0.0	1.942648

Table 7: Best Parameter Settings and Validation Loss

ii. **Model Performance**

Metric	mBART (Science)	mBART (Education)	M2M100 (Science)	M2M100 (Education)
Baseline BLEU Score	0.1223	0.1169	0.0231	0.0347
Fine-Tuned BLEU Score	0.1997	0.1192	0.1200	0.0833
LoRA Optimized BLEU Score	0.1352	0.1153	0.0976	0.0782
LoRA vs Fine-Tuned	-32.30%	-6.41%	-18.67%	-6.12%

Table 8: Comparison of Baseline, Fine-Tuned, and LoRA-Optimized BLEU Scores Across Models and Domains.

iii. **Discussion**
The analysis of percentage decreases in BLEU scores highlights the varying impact of LoRA optimization across models and domains. In the science domain, mBART experienced the largest drop (32.30%), underscoring the challenges of handling complex scientific text. In contrast, M2M100 showed a moderate decline (18.67%), suggesting LoRA's relative efficiency in multilingual contexts. In the education domain, both models exhibited minimal decreases (mBART: 6.41%, M2M100: 6.12%), indicating that LoRA performs well with structured and predictable content. Overall, while LoRA optimization maintains competitive performance, the science domain demonstrates a greater reliance on fine-tuning compared to the education domain.

4.2.2 Incorporating Layer Freezing

i. **Hyperparamer Tuning**
For this technique, we first analyzed the architecture and the number of layers in the encoder and decoder blocks for both models. Since both models are pre-trained for machine translation tasks, we experimented with freezing all but one, all but two, and all but three layers on both the encoder and decoder sides, recording the results.

Category	Model	Encoding Layers Frozen	Decoding Layers Frozen	Loss (Eval)
Education	mBART	8	8	1.9311
Education	M2M	8	8	2.7466
Science	mBART	8	8	1.9126
Science	M2M	8	8	2.6673

Table 9: Best layer freezing parameters and evaluation loss for mBART and M2M models in Education and Science.

ii. **Model Performance**

Metric	mBART (Science)	mBART (Education)	M2M100 (Science)	M2M100 (Education)
Baseline BLEU	0.1223	0.1169	0.0231	0.0347
Fine-Tuned BLEU	0.1997	0.1192	0.1200	0.0809
LoRA BLEU	0.1352	0.1153	0.0976	0.0782
Freezing BLEU	0.1597	0.1178	0.1118	N/A
LoRA vs Fine-Tuned (%)	-32.30%	-6.41%	-18.67%	-6.12%
Freezing vs Fine-Tuned (%)	-20.07%	-4.38%	-6.83%	N/A
Freezing vs LoRA (%)	18.10%	2.17%	14.61%	N/A

Table 10: Comparison of BLEU Scores and Performance Percentages Across Models and Domains

Comparing The Models Sofar:

(a) **Layer Freezing vs. LoRA:**

- *Performance Comparison:* Layer freezing consistently achieves higher BLEU scores than LoRA, with improvements ranging from **2.17%** to **18.10%**.
- *Parameter Analysis:* Layer freezing enables a substantially larger set of trainable parameters than LoRA:
  - For **mBART**, layer freezing offers **375.74M** trainable parameters compared to LoRA's **76.75M**.
  - For **MTM**, layer freezing offers **248.76M** trainable parameters compared to LoRA's **76.75M**.

(b) **LoRA vs. Fine-Tuned:**

- *Performance Comparison:* LoRA consistently yields lower BLEU scores compared to a fully fine-tuned model. The observed performance drops range from **-6.12%** to **-32.30%**, which can be attributed to the substantial reduction in trainable parameters.
- *Parameter Analysis:* LoRA significantly reduces the number of trainable parameters:
  - For **mBART**, only **76.75M** out of **612M** parameters (12.54%) are trainable.
  - For **MTM**, only **76.75M** out of **485M** parameters (15.82%) are trainable.

(c) **Layer Freezing vs. Fine-Tuned:**

- *Performance Comparison:* Although layer freezing outperforms LoRA in all tested domains, it still falls slightly short of the fully fine-tuned model. The performance reductions range from **-4.38%** to **-20.07%**.
- *Parameter Analysis:* By freezing 8 of the 12 layers, parameter updates are limited to a subset of the model:
  - For **mBART**, trainable parameters are reduced to **375.74M** out of **610M** (61.51%).
  - For **MTM**, trainable parameters are reduced to **248.76M** out of **483M** (51.41%).

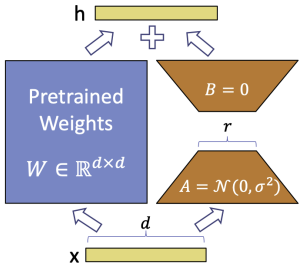

Figure 3: We only train A and B. LoRA Illustration from Paper [3]

GPU Usage Comparisons:

All GPU memory usage was measured on an NVIDIA A100 with 40 GB VRAM available through Google Colab, using PyTorch version `1.12.1` and CUDA version `11.3`. Measurements were taken using NVIDIA's `nvidia-smi` tool during the peak memory consumption phase of training.

- **Normal Fine-Tuning:**
  - Using all parameters results in the highest GPU memory usage, approximately **7 GB** for mBART and **5.5 GB** for M2M100.
  - This approach also incurs considerably longer training times compared to the other methods.
- **LoRA:**
  - This method is the most parameter-efficient, requiring only around **2 GB** of memory for both models.
  - It also maintains stable memory usage throughout the training process, making it ideal for resource-constrained settings.
- **Layer Freezing:**
  - Freezing roughly half the parameters reduces memory usage to approximately **4.5 GB** for mBART and **3.5 GB** for M2M100.
  - Training times are shorter than normal fine-tuning and are comparable to those observed with LoRA.

4.2.3 Exploring Quantization:

In the previous sections, we explored multi-language machine translation models and various fine-tuning techniques. However, we now aim to evaluate whether large quantized general-purpose models and compact, language-specific models can remain competitive in translation tasks while being small enough to run on personal computers.

To this end, we used two models:

1. **Chinese-Alpaca**: A fine-tuned version of LLaMA with 7 billion parameters, optimized for Chinese content. Its quantized version, using a 4-bit format, reduces storage requirements to 4 GB, making it suitable for personal computers.
2. **Marian MT**: A compact, language-specific Chinese-to-English translation model with 60 million parameters, designed exclusively for this language pair.

Model Name	Trainable Parameters	Education BLEU Score	Science BLEU Score
Helsinki (Marian MT)	77,419,008	0.1029	0.1058
TheBloke/Chinese -Alpaca-2-7B-GPTQ	453,251,072	0.0511	0.0403

Table 11: Trainable Parameters and BLEU Scores for Helsinki (Marian MT) and TheBloke/Chinese-Alpaca-2-7B-GPTQ across Education and Science domains.

From our results above, we notice that quantized models offer significant storage and inference efficiency benefits. However, their BLEU scores demonstrate the inherent trade-off between computational efficiency and performance. Quantization makes large models accessible for personal use but falls short in competitive translation tasks, especially when compared to more sophisticated techniques like layer freezing or LoRA.

Comparison of Parameters			
Model and Configuration	Total Parameters	Trainable Parameters (All)	Percentage to Fine-tune
mBART (original)	610,879,488	610,879,488	100.00%
mBART (LoRA)	612,059,136	76,750,848	12.54%
mBART (Layer Freezing)	610,879,488	375,736,320	61.51%
MTM (original)	483,905,536	483,905,536	100.00%
MTM (LoRA)	485,085,184	76,750,848	15.82%
MTM (Layer Freezing)	483,905,536	248,762,368	51.41%
Helsinki (Marian MT)	77,419,008	77,419,008	100.00%
TheBloke/Chinese-Alpaca-2-7B-GPTQ	453,251,072	453,251,072	100.00%

Table 12: Comparison of Parameters Across Models and Configurations

The above table highlights the trade-offs be-

tween the fine-tuning techniques implemented and their impact on computational efficiency:

- **Original Models:** Both mBART and MTM fully fine-tune all parameters (100%), achieving the highest performance but at significant computational cost.
- **LoRA:** Drastically reduces trainable parameters to 12.54% (mBART) and 15.82% (MTM), offering computational efficiency.
- **Layer Freezing:** Balances efficiency and performance by freezing lower layers, with 61.51% (mBART) and 51.41% (MTM) of parameters trainable.
- **Compact Models:** Helsinki (Marian MT) utilizes all of its 77M parameters, benefiting from its smaller size, while the quantized Chinese-Alpaca (GPTQ) retains 453M trainable parameters. Despite the larger parameter count, the reduced precisions (4-bit and 8-bit) significantly lower computational requirements, making them highly efficient for resource-constrained environments.

4.3 Error Analysis

We analyzed errors across mBART and M2M100 models under the following methods: Base, Fine-Tuned, Freeze, and LoRA on the Education Domain.

4.3.1 Classification of Errors

i. **Word-Level Errors** Issues with word choice, missing words, or extra words.
Example:
   - **Ground Truth:** *Though he is fifteen, he has a mental age of less than five.*
   - **Prediction:** *Although he is 15 years old, his intellectual age is less than 5 years old.*

ii. **Structural Errors** Sentence-level inconsistencies, such as grammar issues or word order mismatches.
Example:
   - **Ground Truth:** *And few companies offer more products for the management, conversion, distribution and minimization of power than Fairchild.*
   - **Prediction:** *From the beginning to the end we adhere to one strategy: to become the world's leading provider of high performance products for many markets.*

iii. **Other Errors** Rare issues like nonsensical or untranslated outputs (likely translate to other languages).
Example:
   - **Ground Truth:** *Simplify the axiom system of lattice implication algebras, which was given by Y.*
   - **Prediction:** *与えられた代数軸系を含む代数系があり、別の軸系を提示します。*

4.3.2 Summary of Error Trends

M2M100 Models:

- **LoRA:** Highest Word-Level Errors (3,960), with frequent missing/substituted words and structural misalignments (2,057).
- **Fine-Tuned:** High Word-Level Errors (3,884) and Structural Errors (2,085), reflecting unstable sentence structures.
- **Base/Freeze:** Moderate Structural Errors (˜1,500–1,900); dominated by Word-Level Errors.

mBART Models:

- **LoRA/Fine-Tuned:** Lower Word-Level Errors (˜3,250), occasional synonyms or word omissions; minimal Structural Errors (˜1,300).
- **Base/Freeze:** Stable grammar with fewer structural inconsistencies (˜1,200); errors mainly in word substitutions.

5 Conclusions

This project underscores the potential of parameter-efficient fine-tuning techniques in adapting MT models for edge device deployment. By systematically comparing traditional fine-tuning with methods like LoRA and layer freezing, we revealed trade-offs between computational efficiency and translation quality. The findings highlight the viability of lightweight MT models, especially in domains with predictable content, while demonstrating the need for further research in complex contexts, such as scientific translations. Future work will focus on integrating quantization techniques and exploring domain-specific optimization to further enhance the practicality and robustness of MT systems in constrained environments.

- **Fine-Tuned:** Fully utilizes all parameters (100%) for maximum performance.

- **LoRA:** Trades performance for computational efficiency by updating only 12–16% of parameters.
- **Layer Freezing:** Strikes a balance, freezing lower layers to maintain efficiency while training the more adaptable upper layers, outperforming LoRA in most scenarios.
- **Quantization:** Significantly decreases memory usage and enables deployment on resource-limited devices, but often comes at the cost of reduced translation quality, especially in complex domains like science.

6 Acknowledgments

We extend our deepest gratitude to our mentor and teaching assistant for the course, Ugurcan Vurgun, for his invaluable guidance and support throughout this project. We also acknowledge Professor Mark Yatskar for his inspiring lectures and mentorship, which have profoundly shaped our understanding of the subject. Additionally, we thank the creators of the UM-Corpus dataset and hugging-face for their contributions to advancing research in machine translation. Finally, we are grateful to our peers for their insightful feedback and collaboration, as well as our institution for providing the resources necessary to conduct this study.

7 Appendix

7.1 Hyper-parameter Tuning for Baseline Model Results

Learning Rate	Batch Size	Number of Epochs	Training Loss	Validation Loss
2e-5	8	1	1.9563	1.9182
2e-5	8	2	1.4845	1.9859
2e-5	8	3	1.2150	2.0565
2e-5	16	1	1.9466	1.9152
2e-5	16	2	1.5788	1.9522
2e-5	16	3	1.3677	1.9965
3e-5	8	1	1.9760	1.9337
3e-5	8	2	1.3826	2.0395
3e-5	8	3	1.0099	2.1650
3e-5	16	1	1.9566	1.9234
3e-5	16	2	1.4807	1.9864
3e-5	16	3	1.1861	2.0703
5e-5	8	1	2.0179	1.9713
5e-5	8	2	1.2685	2.1388
5e-5	8	3	0.7508	2.3698
5e-5	16	1	1.9842	1.9463
5e-5	16	2	1.3523	2.0622
5e-5	16	3	0.9291	2.2199

Table 13: mBART Fine-Tuning Hyperparameters and Losses for Education

Learning Rate	Batch Size	Number of Epochs	Training Loss	Validation Loss
2e-5	8	1	1.8151	1.8982
2e-5	8	2	1.3700	1.8215
2e-5	8	3	1.1274	1.8953
2e-5	16	1	1.9304	1.8253
2e-5	16	2	1.5488	1.8156
2e-5	16	3	1.2680	1.8524
3e-5	8	1	1.9062	1.8168
3e-5	8	2	1.2550	1.8457
3e-5	8	3	0.9232	1.9832
3e-5	16	1	1.9273	1.8188
3e-5	16	2	1.4352	1.8282
3e-5	16	3	1.0832	1.9074
5e-5	8	1	1.9328	1.8386
5e-5	8	2	1.1167	1.9054
5e-5	8	3	0.6745	2.3698
5e-5	16	1	1.9416	1.9463
5e-5	16	2	1.2848	2.0622
5e-5	16	3	0.8313	2.2199

Table 14: mBART Fine-Tuning Hyperparameters and Losses for Science

Learning Rate	Batch Size	Number of Epochs	Training Loss	Validation Loss
2e-5	8	1	2.6341	2.5277
2e-5	8	2	2.1900	2.4427
2e-5	8	3	1.8121	2.4457
2e-5	16	1	2.6846	2.5577
2e-5	16	2	2.2645	2.4678
2e-5	16	3	1.9541	2.4428
3e-5	8	1	2.6137	2.5050
3e-5	8	2	2.0599	2.4494
3e-5	8	3	1.6219	2.4748
3e-5	16	1	2.6547	2.5285
3e-5	16	2	2.1253	2.4423
3e-5	16	3	1.7659	2.4484
5e-5	8	1	2.6147	2.5022
5e-5	8	2	1.9222	2.4868
5e-5	8	3	1.3821	2.5506
5e-5	16	1	2.6320	2.5020
5e-5	16	2	1.9724	2.4582
5e-5	16	3	1.5227	2.4989

Table 15: M2M100 Fine-Tuning Hyperparameters and Losses for Education

Learning Rate	Batch Size	Number of Epochs	Training Loss	Validation Loss
2e-5	8	1	1.5302	1.5447
2e-5	8	2	1.1101	1.5898
2e-5	8	3	0.7915	1.6707
2e-5	16	1	1.4962	1.5333
2e-5	16	2	1.1709	1.5618
2e-5	16	3	0.9250	1.6162
3e-5	8	1	1.5792	1.5947
3e-5	8	2	1.0303	1.6540
3e-5	8	3	0.6408	1.7778
3e-5	16	1	1.5286	1.5670
3e-5	16	2	1.0864	1.6112
3e-5	16	3	0.7767	1.6992
5e-5	8	1	1.6798	1.6967
5e-5	8	2	0.9531	1.7703
5e-5	8	3	0.4760	1.9640
5e-5	16	1	1.5986	1.6399
5e-5	16	2	0.9914	1.7082
5e-5	16	3	0.5911	1.8461

Table 16: M2M100 Fine-Tuning Hyperparameters and Losses for Science

7.2 Hyper-parameter Tuning for LoRA Results

Rank Values (r)	Scaling Factor (α)	Dropout Values	Training Loss	Validation Loss
8	16	0.0	2.269500	2.186194
8	16	0.1	2.218100	2.136166
8	16	0.2	2.228000	2.146366
8	32	0.0	2.072600	2.003432
8	32	0.1	2.070800	2.004421
8	32	0.2	2.072200	2.005456
8	**64**	**0.0**	**2.033000**	**1.967467**
8	64	0.1	2.031000	1.968252
8	64	0.2	2.032000	1.968983
16	16	0.0	2.280500	2.197836
16	16	0.1	2.280000	2.202169
16	16	0.2	2.290800	2.214560
16	32	0.0	2.076000	2.006214
16	32	0.1	2.073900	2.007113
16	32	0.2	2.075300	2.008230
16	64	0.0	2.039500	1.973194
16	64	0.1	2.036900	1.973644
16	64	0.2	2.037400	1.973854
32	16	0.0	2.260900	2.176215
32	16	0.1	2.266600	2.187254
32	16	0.2	2.276200	2.197995
32	32	0.0	2.075400	2.005779
32	32	0.1	2.073200	2.006605
32	32	0.2	2.074700	2.007794
32	64	0.0	2.041200	1.974903
32	64	0.1	2.038600	1.975254
32	64	0.2	2.039600	1.975913

Table 17: Hyperparameter Fine-Tuning for LoRA-injected mBART on Education Domain

Rank Values (r)	Scaling Factor (α)	Dropout	Training Loss	Validation Loss
8	16	0.0	2.075400	2.084480
8	16	0.1	2.073500	2.084773
8	16	0.2	2.074500	2.085670
8	32	0.0	2.045600	2.054320
8	32	0.1	2.044000	2.054817
8	32	0.2	2.045200	2.055982
8	**64**	**0.0**	**2.016100**	**2.024966**
8	64	0.1	2.015000	2.025520
8	64	0.2	2.016400	2.026753
16	16	0.0	2.075600	2.084319
16	16	0.1	2.073300	2.084745
16	16	0.2	2.074400	2.085694
16	32	0.0	2.045800	2.054456
16	32	0.1	2.043700	2.054771
16	32	0.2	2.045100	2.055901
16	64	0.0	2.016300	2.025228
16	64	0.1	2.014800	2.025674
16	64	0.2	2.016100	2.026709
32	16	0.0	2.075500	2.084476
32	16	0.1	2.073600	2.085136
32	16	0.2	2.074700	2.086110
32	32	0.0	2.046100	2.055015
32	32	0.1	2.044500	2.055676
32	32	0.2	2.045600	2.056716
32	64	0.0	2.016400	2.025542
32	64	0.1	2.015300	2.026190
32	64	0.2	2.016600	2.027361

Table 18: Hyperparameter Fine-Tuning for LoRA-injected mBART on Science Domain

Rank Values (r)	Scaling Factor (α)	Dropout	Training Loss	Validation Loss
8	16	0.0	3.061000	2.842442
8	16	0.1	3.059800	2.842457
8	16	0.2	3.061900	2.844099
8	32	0.0	3.031200	2.816317
8	32	0.1	3.032200	2.817689
8	32	0.2	3.034500	2.819312
8	**64**	**0.0**	**2.999900**	**2.791431**
8	64	0.1	3.003500	2.794098
8	64	0.2	3.006600	2.796152
16	16	0.0	3.061600	2.842710
16	16	0.1	3.062200	2.844057
16	16	0.2	3.063800	2.845312
16	32	0.0	3.032000	2.816813
16	32	0.1	3.033200	2.818718
16	32	0.2	3.035100	2.820055
16	64	0.0	3.000700	2.792005
16	64	0.1	3.003600	2.794752
16	64	0.2	3.006200	2.796460
32	16	0.0	3.061600	2.842674
32	16	0.1	3.061800	2.843768
32	16	0.2	3.063400	2.844864
32	32	0.0	3.032100	2.816821
32	32	0.1	3.032900	2.818080
32	32	0.2	3.034800	2.819568
32	64	0.0	3.001200	2.792586
32	64	0.1	3.003700	2.794915
32	64	0.2	3.006200	2.796576

Table 19: Hyperparameter Fine-Tuning for LoRA-injected M2M100 on Education Domain (3rd epoch)

Rank (r)	Alpha (α)	Dropout	Training Loss	Validation Loss	Evaluation Loss
8	16	0.0	3.233600	3.063320	3.063320
8	16	0.1	3.227100	3.056787	3.056787
8	16	0.2	3.237200	3.066366	3.066366
8	32	0.0	3.094100	2.931028	2.931028
8	32	0.1	3.096200	2.934254	2.934254
8	32	0.2	3.101000	2.938328	2.938328
8	**64**	**0.0**	**3.033500**	**2.874502**	**2.874502**
8	64	0.1	3.033800	2.876885	2.876885
8	64	0.2	3.037700	2.880006	2.880006
16	16	0.0	3.240600	3.070621	3.070621
16	16	0.1	3.249300	3.079134	3.079134
16	16	0.2	3.260000	3.089163	3.089163
16	32	0.0	3.098900	2.936606	2.936606
16	32	0.1	3.101100	2.940071	2.940071
16	32	0.2	3.106300	2.944297	2.944297
16	64	0.0	3.036000	2.877845	2.877845
16	64	0.1	3.036300	2.879981	2.879981
16	64	0.2	3.040700	2.883339	2.883339
32	16	0.0	3.244200	3.073837	3.073837
32	16	0.1	3.251700	3.081498	3.081498
32	16	0.2	3.263600	3.092737	3.092737
32	32	0.0	3.102300	2.939461	2.939461
32	32	0.1	3.103100	2.942031	2.942031
32	32	0.2	3.108300	2.946282	2.946282
32	64	0.0	3.035800	2.877304	2.877304
32	64	0.1	3.035900	2.879772	2.879772
32	64	0.2	3.040700	2.883350	2.883350

Table 20: Hyperparameter Fine-Tuning for LoRA-injected M2M100 on Science Domain

7.3 GPU Consumption Graphs

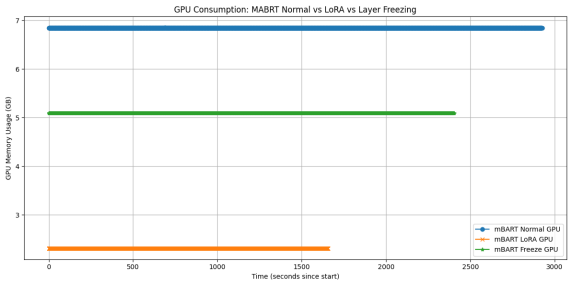

Figure 4: GPU Consumption: mBART Normal vs LoRA vs Layer Freezing on Education Domain

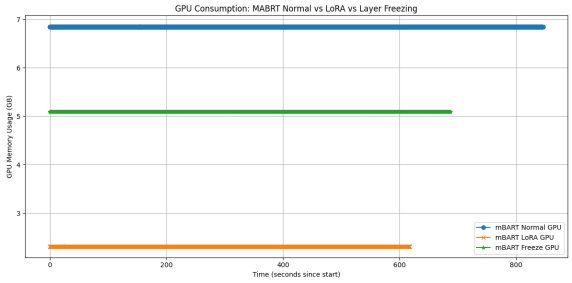

Figure 5: GPU Consumption: mBART Normal vs LoRA vs Layer Freezing on Science Domain

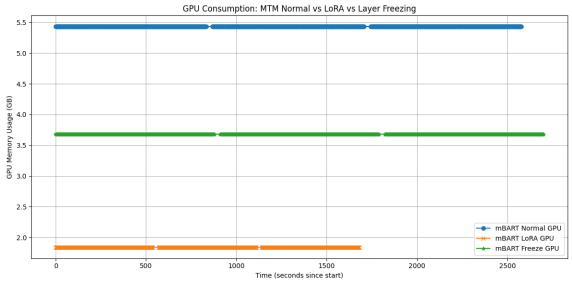

Figure 6: GPU Consumption: M2M Normal vs LoRA vs Layer Freezing on Education

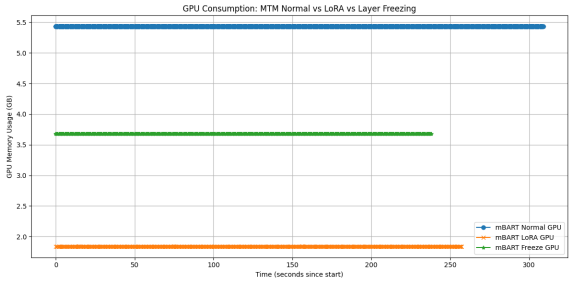

Figure 7: GPU Consumption: M2M Normal vs LoRA vs Layer Freezing on Science Domain

References

[1] Xiaoyu Chen, Yafu Li, and Zhaopeng Tu. "Exploring Robustness of Machine Translation Metrics: A Study of Twenty-Eight Automatic Metrics in the WMT22 Metric Task". In: *Proceedings of the Seventh Conference on Machine Translation (WMT)*. 2022.

[2] Xiaoyu Chen, Heng Wang, and Zhaopeng Tu. "Multifaceted Challenge Set for Evaluating Machine Translation Performance". In: *Proceedings of the WMT23 Test Suites Shared Task*. 2023.

[3] Edward J. Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations (ICLR)*. 2022.

[4] Kishore Papineni et al. "BLEU: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2002, pp. 311–318.

[5] Maja Popović. "chrF: character n-gram F-score for automatic MT evaluation". In: *Proceedings of the Tenth Workshop on Statistical Machine Translation (WMT)*. 2015, pp. 392–395.

[6] Liang Tian et al. "UM-Corpus: A Large English-Chinese Parallel Corpus for Statistical Machine Translation". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*. 2014, pp. 1837–1842.

[7] Tianyi Zhang et al. "BERTScore: Evaluating Text Generation with BERT". In: *International Conference on Learning Representations (ICLR)*. Originally published as arXiv:1904.09675 [cs.CL]. 2020.