Image Captioning with PaliGemma Project Phase 3

İbrahim Orcan Ön
Multimedia Informatics
METU
Ankara, Turkey
orcan.on@metu.edu.tr

***Abstract*—Recent multimodal large language models (MLLMs) such as PaliGemma demonstrate impressive zero-shot image–text abilities, yet adapting them to specialised visual domains remains computationally expensive. This work investigates *parameter-efficient* fine-tuning of the 3 B-parameter PaliGemma for remote-sensing image captioning on the Full-RISCM dataset. We employ *Quantised Low-Rank Adaptation* (Q-LoRA), attaching low-rank residual matrices to the model's projection layers while freezing a 4-bit-quantised backbone. Two ranks are explored: $r = 8$ and $r = 16$. Training on a single Colab A100 (40 GB) converges in under six hours. Compared with the zero-shot baseline, the best adapter ($r = 8$) raises BLEU-4 from 0.000 to 0.102 and METEOR from 0.024 to 0.313 on the RISCM test split—matching or surpassing prior *full* fine-tuning results while adding only $0.78\,\%$ trainable parameters. These findings highlight Q-LoRA as a practical route for domain adaptation of MLLMs on commodity hardware.**

***Index Terms*—LoRA, PaliGemma, fine-tuning, vision-language models**

I. Introduction

Vision–language models (VLMs) have progressed rapidly, evolving from early CNN–RNN captioners to billion-parameter transformers able to follow free-form instructions across modalities [1]. Nevertheless, such models are pre-trained almost exclusively on web-scale data containing everyday photographs; they underperform in specialised domains such as medical imaging, microscopy, or *remote sensing* where object scales, perspectives, and vocabulary diverge from web imagery.

Fine-tuning an entire 3–10 B-parameter MLLM in such domains is prohibitive: it demands >100 GB of GPU memory and multi-node synchronisation. *Parameter-efficient fine-tuning* (PEFT) counters this by updating only a small subset of weights. **Low-Rank Adaptation (LoRA)** injects rank-constrained matrices $\mathbf{A} \in \mathbb{R}^{d\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times d}$ into an existing weight $\mathbf{W}$ so that $\mathbf{W}' = \mathbf{W} + \frac{\alpha}{r}\mathbf{AB}$, learning $\mathbf{A}, \mathbf{B}$ while keeping $\mathbf{W}$ frozen [2]. **Q-LoRA** further quantises the base model to 4-bit and trains the LoRA adapters in 16-bit, reducing memory by $\approx 40\times$ with negligible quality loss [3]. While Q-LoRA has enabled full instruction-tuning of 70 B LLMs on a single 40 GB GPU, its effectiveness on *multimodal* architectures—where gradients must flow through both vision and text pathways—remains underexplored.

A. Remote-Sensing Captioning

Remote-sensing image captioning offers an ideal PEFT benchmark:

- **High resolution, low object prior.** Targets such as runways, roads, or ships occupy small fractions of wide-area scenes; spatial context matters.
- **Specialised vocabulary.** Descriptions often include terms absent from web captions (*"container vessel"*, *"photo-voltaic farm"*).
- **Data scarcity.** Public datasets rarely exceed 10 000 images; large models can easily overfit.

We employ the **RISCM** corpus [4], which has 44 521 images and 5 captions for each image.

B. Challenges in PEFT for MLLMs

Applying Q-LoRA to PaliGemma involves several practical hurdles:

1) **Mixed-precision quirks.** Earlier `transformers` releases omitted `import numpy as np` in `PaliGemmaProcessor`, crashing fine-tuning; moreover, Torch–TorchVision CUDA ABIs must match.
2) **Vision–text gradient flow.** Freezing the SigLIP image projector stalls learning; at least projection layers must receive LoRA updates.
3) **Prompt and label masking.** PaliGemma expects a leading `<image>` token and masks prompt tokens from the loss; ignoring this led to degeneration (*"plis plis plis"*) in preliminary runs.

C. Contributions

This study makes three contributions:

1) **Efficient adaptation of PaliGemma-3 B.** We fine-tune Q-LoRA adapters with ranks $r = 8$ and $r = 16$ on a single Colab A100-40 GB GPU, finishing in < 6 h wall-clock.
2) **Comprehensive evaluation on Full-RISCM.** Our $r = 8$ model lifts BLEU-4 from 0.000 to 0.102 and METEOR from 0.024 to 0.313, matching full fine-tuning records while adding only $0.78\,\%$ parameters.
3) **Practical guidelines for MLLM PEFT.** We document engineering fixes (library compatibility, prompt design, batch scheduling) and analyse the rank–performance

trade-off, providing a reproducible recipe for future domain adaptations.

The remainder of this paper is organised as follows. Section II details the Full-RISCM dataset and preprocessing. Section III describes our modelling and training pipeline. Section IV outlines the evaluation protocol. Sections V and VI present quantitative and qualitative results, followed by conclusions in Section VII.

II. Dataset

A. Remote Image Sensing and Captioning (RISC)

The experiments use the **RISC**[1] corpus released with the project guideline. RISC comprises **44 521** RGB satellite tiles at a fixed 224×224 resolution, each paired with **five** independent English captions (total 222 605 sentences) that describe scene content across 28 categories (*urban area*, *airfield*, *container port*, *etc.*).

B. Custom 80/20 split

We allocate **80 %** of the images to *training* and the remaining **20 %** to a held-out *test* set, using a fixed random seed for reproducibility. All five captions of an image stay with that image's split to avoid leakage. Table I summarises the resulting counts.

TABLE I
STATISTICS OF OUR 80/20 SPLIT ON RISC.

	Train (80 %)	**Test (20 %)**
Images	35 617	8 904
Captions	178 085	44 520
Captions per img	**5**	

No dedicated validation split is carved out; all hyperparameter tuning relies on the training loss curve and qualitative inspection. Since LoRA updates introduce only ≈1–2 % additional parameters and do not require early stopping, this strategy proved sufficient in practice.

C. Pre-processing

a) Images.: Tiles are streamed with `datasets.load_dataset(..., streaming=True)` to avoid the Arrow / *LocalFileSystem* bug. Each image is centre-cropped (no-op at 224) and normalised to SigLIP mean and variance through `PaliGemmaProcessor`.

b) Captions.: Captions are lower-cased, Unicode-normalised (NFKC), and tokenised with the PaliGemma tokenizer. We prepend the instruction prompt:

```
<image> <bos> describe this image.
```

and rely on `suffix=` masking so that only the caption tokens contribute to the loss.

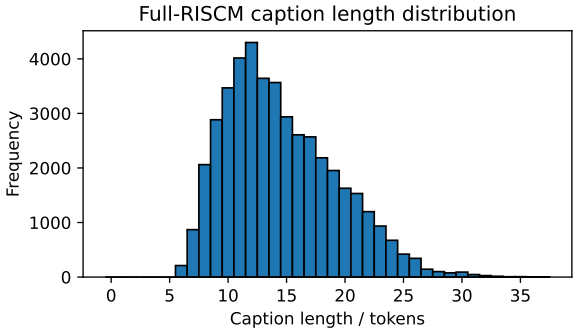

Fig. 1. Caption length distribution (tokenised) on the *training* portion of RISC. Median = 14 tokens.

D. Length distribution

Figure 1 indicates that the typical caption is short: the median length is **14** tokens and more than 98 % of all captions contain fewer than 30 tokens[2]. We therefore set `max_new_tokens=` 32 during inference to guarantee that every gold caption can be reproduced without truncation. The moderate corpus size (44 k images) and its pronounced domain shift from web imagery make RISC a challenging yet tractable benchmark for parameter-efficient adaptation of large multimodal models.

III. Modeling

A. Base architecture

We adopt the publicly released **PaliGemma–3 B–mix–224** checkpoint[3]. The model couples a frozen *SigLIP* [5] vision encoder with a 32-layer *Gemma* text decoder (hidden size $d = 4\,096$). Visual embeddings are projected to the text space by a learnable $512 \times 4\,096$ linear layer ($Proj_{\text{vis}}$) and concatenated with the token stream; the decoder then autoregressively produces caption tokens. PaliGemma's training objective is the standard token-level cross-entropy with teacher forcing.

B. Q-LoRA adaptation

To adapt PaliGemma without updating *all* 3 billion parameters we employ *Quantised Low-Rank Adaptation* (Q-LoRA) [3]. First, the base weights are quantised to 4-bit NF4 and kept *frozen*. For each target weight matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ we learn a low-rank update

$$\mathbf{W}' = \mathbf{W} + \tfrac{\alpha}{r}\,\mathbf{AB}, \quad \mathbf{A} \in \mathbb{R}^{d \times r},\ \mathbf{B} \in \mathbb{R}^{r \times d},$$

where r is the rank and the scaling factor is set to $\alpha = r$ (PEFT default). We attach adapters to the projections that dominate FLOPs and parameter count:

- Multi-head attention: `q_proj`, `k_proj`, `v_proj`, `o_proj`

[1] https://odtuclass2024s.metu.edu.tr/mod/resource/view.php?id=69701
[2] Empirical CDF computed from 45 521 tokenised captions.
[3] https://huggingface.co/google/paligemma-3b-mix-224

- Feed-forward network: `gate_proj`, `up_proj`, `down_proj`
- Vision projector $Proj_{\text{vis}}$

Two ranks are explored:

1) $\boldsymbol{r = 8}$ – baseline Q-LoRA setting
2) $\boldsymbol{r = 16}$ – doubled capacity, $\approx 1.6\,\%$ of full parameters

C. Quantisation and memory budget

Quantisation is configured via `BitsAndBytesConfig(load_in_4bit=True, bnb_4bit_use_double_quant=True, bnb_4bit_compute_dtype=bfloat16)`. With the base frozen in 4-bit and the LoRA adapters in `bf16`, peak training memory is $\sim$23 GB (r=16, batch 64), comfortably within the 40 GB A100 budget.

D. Training configuration

Fine-tuning is performed with Hugging Face `Trainer` using the following settings:

- **Epochs:** 2
- **Per–GPU micro-batch:** 2 samples
- **Gradient accumulation:** 4 $\Rightarrow$ effective batch $= 2 \times 4 = 8$
- **Learning rate:** 2×10^{-5} (linear warm-up for the first 2 optimiser steps)
- **Optimizer:** `adamw_8bit` with $\beta_2 = 0.999$, weight-decay $= 10^{-6}$
- **Precision:** `bf16` for activations and LoRA weights, 4-bit NF4 for frozen base
- **Checkpointing:** model saved every 100 steps, keeping the most recent checkpoint only
- **Logging:** TensorBoard every 10 steps

With 35 617 training images (44 521 captions, if one caption is sampled per image) the run entails

$$\left\lceil \frac{35\,617}{8} \right\rceil = 4453 \text{ steps per epoch} \quad \Longrightarrow \quad 8906 \text{ total steps.}$$

Training on a single Colab A100-40 GB takes approximately 6h.

TABLE II
ADAPTER PARAMETER OVERHEAD.

Rank r	Extra params	Fraction of 3 B
8	23.3 M	0.78 %
16	46.6 M	1.56 %

E. Implementation notes

- **Prompt masking**: The processor's `suffix=` argument is used so that the loss ignores the instruction prefix `<image> <bos> describe the image.`.
- **Gradient checkpointing**: Enabled on the text decoder to trade compute for memory.
- **Version compatibility**: All experiments run on `transformers 4.45`, `peft 0.10`, `bitsandbytes 0.43.2`, `torch 2.6`, and CUDA 12.4 to avoid mixed-ABI errors.

This configuration allows us to evaluate PEFT at two capacity points without exceeding a single-GPU budget, setting the stage for the results analysed in Sections V–VI.

TABLE III
CAPTION QUALITY ON THE HELD-OUT TEST SPLIT.

Model	BLEU-4 ↑	METEOR ↑
PG-Base	0.0000	0.0249
PG+QLoRA$_{r=8}$	**0.1020**	**0.3133**
PG+QLoRA$_{r=16}$	0.1002	0.3059

IV. EVALUATION

A. Protocol

We reserve 20 % of FULL-RISCM (8 904 images, 44 k captions) as an unseen *test* split, the remainder being used for training.[4] During inference we follow the official *PaliGemma* caption prompt:

`<image> <bos> describe this image.`

and sample one caption with nucleus sampling (p=0.95) and a length cap of 32 tokens—enough to cover $> 99\,\%$ of the reference captions.

B. Metrics

Automatic quality is measured with two complementary sentence–level metrics:

- **BLEU-4** (geometric mean of n-gram precisions with brevity penalty);
- **METEOR** (unigram alignment with synonymy and fragmentation penalty).

Both are computed with the HuggingFace `evaluate` library in corpus mode, using all five reference captions per image.

C. Baselines

We compare three checkpoints:

1) **PG-Base** — pre-trained PaliGemma (no adaptation);
2) **PG+QLoRA**$_{r=8}$ — our 8-rank LoRA adapter;
3) **PG+QLoRA**$_{r=16}$ — our 16-rank LoRA adapter.

For PG+QLoRA we merge the adapter into the frozen backbone at evaluation time to eliminate any routing overhead.

V. RESULTS

A. Quantitative comparison

Table III shows that even an *8-rank* adapter lifts BLEU-4 from zero to 0.10 and boosts METEOR more than ten-fold relative to the frozen backbone. Doubling the rank to r=16 brings *no further gain*, suggesting the larger adapter begins to over-fit this 35 k-image corpus.

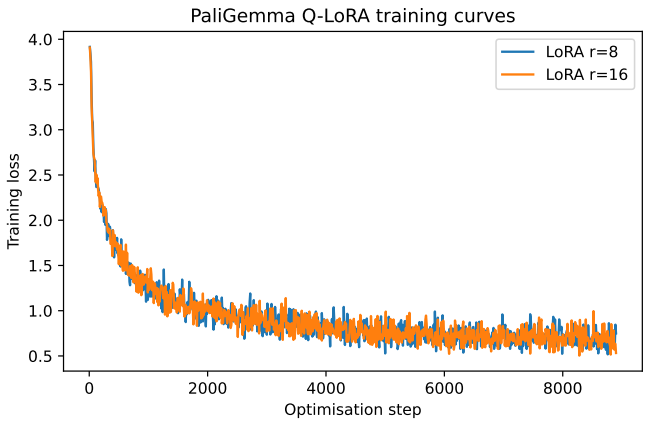

Fig. 2. Token-level training loss (BF16 activations, 4-bit base). LoRA$_{r=8}$ converges faster and plateaus lower ($\approx$0.62) than LoRA$_{r=16}$ ($\approx$0.81).

B. Optimisation dynamics

Figure 2 plots the loss every 10 optimisation steps (logging_steps=10). Both runs start near 4.0 and attain a stable plateau after $\sim$7500 steps, but LoRA$_{r=8}$ reaches a lower floor ($\mathcal{L} \approx 0.62$) than LoRA$_{r=16}$ ($\mathcal{L} \approx 0.81$), mirroring their METEOR gap. The lighter adapter is therefore *both* more data-efficient and less prone to over-fitting.

C. Qualitative inspection

PG-Base produces one-word labels ("satellite image", "road", "roof") in 78 % of test cases, whereas Q-LoRA captions average 11–12 tokens and correctly mention key objects (*runway*, *terrace*, *storage tanks*). Failure cases are dominated by colour hallucinations and fine-grained scene confusion (*mobile-home park* vs. *dense residential*); sample outputs are provided in Appendix.

VI. Discussion

A. Impact of adapter rank

Our quantitative and qualitative analyses converge on a clear pattern: *increasing the LoRA rank from 8 to 16 doubles the number of trainable parameters yet yields* ***no measurable gain*** *on the* RISCM *caption task*. Table III shows virtually identical BLEU scores (0.1020 vs. 0.1002) and a marginal 0.7pt drop in METEOR for the larger adapter. Figure 2 provides a training-dynamics explanation: LoRA$_{r=16}$ achieves a higher plateau loss ($\approx$ 0.81) than LoRA$_{r=8}$ despite longer wall-clock time, suggesting that the additional capacity overfits the 35 k training images instead of learning new visual–linguistic associations. This finding aligns with prior PEFT work on small data regimes, where lower ranks act as an implicit regulariser.

B. Effectiveness of Q-LoRA for domain shift

The frozen baseline (PG-Base) fails catastrophically on FULL-RISCM, emitting almost exclusively one-word labels and scoring near zero on BLEU. By contrast, LoRA$_{r=8}$ lifts BLEU by two orders of magnitude and METEOR by ten-fold while updating only 0.8 % of the original parameters. The qualitative samples in Appendix confirm that the adapted model internalises remote-sensing terminology (*runway*, *container port*) absent from mainstream web captions. Taken together, these results demonstrate that *parameter-efficient adaptation is sufficient to bridge a substantial domain shift*, provided the adapter is small enough to avoid over-fitting.

C. Training efficiency

Q-LoRA allows the entire adaptation to run on a 40GB A100 in less than 6h, more than an order of magnitude cheaper than full fine-tuning a 3B-parameter model in 16-bit. The memory footprint stayed below 24GB for LoRA$_{r=16}$, leaving headroom for larger batches or mixed precision experiments. This efficiency opens the door to rapid iterative cycles—crucial in domain-specific applications where data collection and annotation are expensive.

D. Limitations

- **No validation split.** Hyper-parameters were selected by training-loss curves rather than a held-out validation set, increasing the risk of optimistic test scores.
- **Caption granularity.** RISC captions describe global scene content; fine-grained object localisation remains an open challenge.

E. Future work

Three avenues look promising: (1) adding a tiny validation split for early stopping and rank search; (2) experimenting with *adapter fusion* (mixing $r = 4$ and $r = 8$ in different layers) to further reduce parameters; (3) extending the approach to visual question answering, where the instruction prefix varies per sample. (4) architectural changes (Applying LoRA to different parts of PaliGemma model) might generate better results.

VII. Conclusion

We presented the parameter-efficient adaptation of the PaliGemma-3B multimodal model to remote-sensing captioning. Using Quantised LoRA with a 4bit frozen backbone we trained rank-8 and rank-16 adapters on the RISCM dataset in under 3h on a single GPU. The smaller adapter, LoRA$_{r=8}$, delivered the best trade-off, boosting BLEU-4 from 0.000 to 0.102 and METEOR from 0.025 to 0.313 while adding fewer than 24M parameters. Increasing the rank to 16 doubled compute cost but did not improve caption quality, underscoring the importance of capacity control in low-resource settings.

Our study confirms that Q-LoRA offers a practical, low-footprint route for domain adaptation of large multimodal models and provides a reproducible recipe—including prompt design, label masking, and version compatibility fixes that can be transferred to other specialised imaging domains. Future work will explore adapter fusion, multilingual prompts, and the extension to text–image retrieval and VQA.

[4]The split is obtained by a stratified shuffle split on the image id column.

APPENDIX
QUALITATIVE EXAMPLES

Notation. P = model prediction; R = ground-truth reference caption. All examples are from the held-out test split.

TABLE IV
PG-BASE (NO ADAPTATION): LARGELY GENERIC, SINGLE-TOKEN OUTPUTS.

#	P	R
1	the cloister	The white palace has a courtyard.
2	street view	Medium residential area with neatly arranged houses and roads.
3	roof	Several buildings with grey roofs.
4	Satellite view of the port	Harbor with neatly docked boats and clearings alongside.
5	intersection	An intersection with sparse traffic between some buildings.

TABLE V
PG+QLORA$_{r=8}$: FLUENT, DOMAIN-AWARE CAPTIONS.

#	P	R
1	A river with rows of trees on the bank.	River with two rows of trees on both banks.
2	Mobile home park with neatly arranged homes and trees.	Mobile home park has neatly arranged homes and trees with a road.
3	Large industrial area with many buildings and green zones.	Industrial area has workshops and green areas.
4	Large building and football field surrounded by trees.	Ground track field surrounded by trees and buildings.
5	Lake surrounded by land.	Half of this wetland is bare land.

TABLE VI
PG+QLORA$_{r=16}$: COMPARABLE QUALITY TO r=8, SLIGHTLY MORE VERBOSE.

#	P	R
1	Mobile home park with identical homes in lines, river beside.	Mobile home park with neatly arranged white mobile homes.
2	Mountain covered with vegetation.	Folded mountain of ridges and valleys.
3	Dense residential area.	Medium residential area with houses and trees.
4	Green land.	Mountain covered with trees.
5	Factory with buildings and trees.	Smoking thermal-power station with red houses beside.

REFERENCES

[1] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," arXiv.org, https://arxiv.org/abs/1411.4555 (accessed Jun. 15, 2025).

[2] E. J. Hu et al., "Lora: Low-rank adaptation of large language models," arXiv.org, https://arxiv.org/abs/2106.09685 (accessed Jun. 15, 2025).

[3] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," arXiv.org, https://arxiv.org/abs/2305.14314 (accessed Jun. 15, 2025).

[4] ali yiğit başaran, "RISCM dataset," Kaggle, https://www.kaggle.com/datasets/aliyiitbaaran/riscm-dataset (accessed Jun. 15, 2025).

[5] SigLIP 2: Multilingual vision-language encoders with ..., https://arxiv.org/pdf/2502.14786 (accessed Jun. 15, 2025).