# Few-Shot Image Captioning Report: Modality-Specific Fine-Tuning and Inference Strategies on the Pokémon Dataset

Reman Mahameed , Teba Suh

February 28, 2026

**Abstract**

This report details the adaptation of a pre-trained vision-language model for image captioning using a strictly constrained subset of 700 images from the Pokémon BLIP Captions dataset. We demonstrate that full-parameter fine-tuning on such a limited dataset results in rapid overfitting. To achieve stable generalization, we conduct an ablation study utilizing Low-Rank Adaptation (LoRA) across isolated model modalities. The report identifies a Text-Only LoRA configuration as the most optimal and parameter-efficient training strategy, details its hyperparameter optimization, and outlines the decoding parameters required to generate coherent captions during inference.

## 1 Introduction

Image captioning requires models to successfully map extracted visual features to natural language vocabularies. The objective of this project is to train a model to caption images within a highly specific visual domain: 2D stylized illustrations from a 700-image subset of the Pokémon BLIP Captions dataset. We utilize `microsoft/git-base` (Generative Image2Text), a Transformer-based architecture of approximately 178 million parameters that combines a CLIP-based vision encoder with a BERT-based text decoder.

## 2 Fine-Tuning Strategies and Modality Ablation

To establish an optimal training pipeline, we evaluated full fine-tuning against various Low-Rank Adaptation (LoRA) targeting strategies. LoRA mitigates overfitting by freezing the base model weights ($W_0 \in \mathbb{R}^{d \times k}$) and injecting trainable rank decomposition matrices:

$$W = W_0 + \frac{\alpha}{r} BA$$

where $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times k}$ are low-rank matrices ($r \ll \min(d, k)$), and $\alpha$ is a scaling factor.

### 2.1 Experimental Setup

All experiments were conducted using the Hugging Face `Trainer` API with gradient checkpointing enabled. The shared hyperparameters across all runs were: an AdamW optimizer, a learning rate of $1 \times 10^{-4}$, 8 epochs, a batch size of 2 (with 4 gradient accumulation steps), and FP16 mixed precision.

## 2.2 Ablation Study Results

Multimodal architectures process discrete data types, allowing us to isolate parameter updates to specific components of the model. Table 1 outlines the performance of full fine-tuning against modality-specific LoRA targeting after 8 training epochs.

| **Training Method** | **Target Modules** | **Final Train Loss** | **Final Val Loss** | **Best Val Loss** |
|---|---|---|---|---|
| Full Fine-Tuning | All Parameters | 0.106 | 1.811 | 1.685 (Ep 4) |
| LoRA (Vision Only) | `q_proj, k_proj, v_proj, out_proj` | 2.476 | 2.543 | 2.543 (Ep 8) |
| LoRA (Text Only) | `query, key, value, dense` | 1.415 | 1.704 | **1.704 (Ep 8)** |
| LoRA (Both) | Vision + Text Layers | 0.660 | 1.705 | 1.681 (Ep 4) |

Table 1: Comparison of Fine-Tuning Methods at Epoch 8

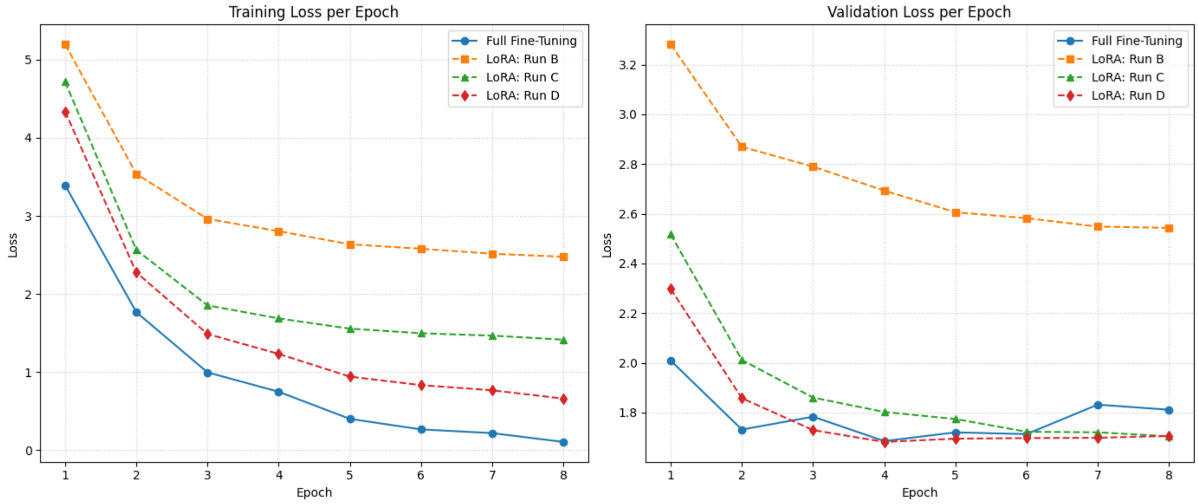


Figure 1: Training and Validation Loss across 8 epochs. The full fine-tuning baseline exhibits severe overfitting (U-shaped validation curve), while the Text-Only and Dual-Encoder LoRA setups maintain stable generalization.

## 2.3 Analysis of Visual Learning Curves

As illustrated in Figure 1, plotting the learning curves provides immediate visual confirmation of the training dynamics:

- **Full Fine-Tuning (Overfitting):** The visual data exhibits textbook overfitting. While the training loss plummeted to nearly zero, the validation loss formed a distinct U-shape, bottoming out at Epoch 4 before steadily rising. This divergence proves the full parameter updates caused the model to memorize the training images rather than learn generalized rules.

- **Vision-Only LoRA (Underfitting):** The validation loss curve remained elevated above 2.50, demonstrating that adapting only visual representations is insufficient. The frozen text decoder lacked the capacity to learn domain-specific vocabulary.

- **Text-Only vs. Dual-Encoder LoRA (The Generalization Gap):** While both configurations achieved nearly identical final validation losses ($\sim$1.70), their training dynamics differed significantly. The Dual-Encoder model drove its training loss down to 0.660, creating a large generalization gap that indicates a tendency to overfit and memorize the training pixels. The Text-Only model maintained a much tighter, healthier gap (Train:

1.415, Val: 1.704), proving it learned generalized rules without memorization. Consequently, Text-Only was selected as the optimal architecture.

# 3 Hyperparameter Optimization for Text-Only LoRA

Having identified the Text-Only configuration as the most efficient architectural strategy, further optimization was required to select the ideal LoRA hyperparameters. We evaluated different configurations of intrinsic rank ($r$) and scaling factor ($\alpha$) to balance representational capacity against the risk of overfitting on the small dataset.

## 3.1 Training Hyperparameters

**Learning rate.** We set the learning rate to $\eta = 5 \times 10^{-5}$. Empirically, this value produced more coherent captions and better validation behavior than smaller learning rates, which tended to under-train the adapter within the same compute budget. Since LoRA trains only a small number of additional parameters (with the backbone frozen), a moderately higher learning rate is often beneficial to achieve effective adaptation from limited data.

**Number of epochs.** We train for 15 epochs to ensure sufficient exposure to the dataset despite its small size. The validation loss decreases rapidly in early epochs and then improves more gradually, indicating that additional epochs still contribute incremental gains. To reduce the risk of selecting an overfit final checkpoint, we monitor validation loss and use best-checkpoint selection (see below).

**Batch size and gradient accumulation.** Due to memory constraints of transformer-based vision–language models, we use a per-device batch size of 2. To obtain a more stable gradient estimate without increasing memory usage, we apply gradient accumulation with 4 steps, yielding an effective batch size of

$$B_{\text{eff}} = 2 \times 4 = 8.$$

This improves optimization stability relative to a raw batch size of 2 while remaining feasible on the available hardware.

**Evaluation and checkpointing strategy.** We evaluate and save checkpoints once per epoch (`eval_strategy="epoch"`, `save_strategy="epoch"`). This is appropriate for a small dataset where each epoch is relatively fast, and it provides frequent feedback on generalization. We also enable `load_best_model_at_end=True`, selecting the checkpoint with the lowest validation loss, which provides an implicit regularization effect by avoiding the last epoch if it begins to overfit.

**Warmup.** We use a warmup ratio of 0.05, meaning the learning rate is linearly increased from 0 to $\eta$ over the first 5% of training steps. Warmup reduces early-training instability when adapting large transformer models.

**Weight decay.** We set weight decay to 0.0. Because LoRA introduces a relatively small set of trainable low-rank parameters while keeping the backbone frozen, strong weight regularization can unnecessarily limit the adapter's ability to capture dataset-specific captioning patterns. In this setting, validation monitoring and best-checkpoint selection provide sufficient control against overfitting.

## 3.2 LoRA Configuration

**Role of the rank $r$.** The rank $r$ controls the capacity of the adaptation. Larger $r$ increases expressiveness (more degrees of freedom) but also increases the number of trainable parameters and the risk of overfitting. Smaller $r$ constrains the update and can improve generalization, but if too small it can underfit and fail to capture domain-specific patterns.

**Role of the scaling factor $\alpha$.** The scaling factor $\alpha$ controls the effective magnitude of the LoRA update through the factor $\alpha/r$. This stabilizes optimization by preventing the low-rank update from becoming disproportionately large or small relative to the frozen base weights.

**LoRA dropout and target modules.** LoRA dropout regularizes the adapter pathway by applying dropout within the LoRA branch during training, which helps reduce overfitting on small datasets. LoRA is injected into selected transformer submodules (commonly attention projections such as query/key/value and/or related projections). Targeting attention-related matrices is effective because it allows the model to adjust how visual features and text context are attended to, while leaving most of the pre-trained backbone unchanged.

**Empirical choice of $(r, \alpha)$.** We compared multiple LoRA capacities by varying the rank $r$ while monitoring validation loss and qualitative caption quality. Holding $\alpha = 32$ fixed, we observed that $r = 16$ consistently produced better captions and better validation behavior than $r = 32$. With only 700 images, the higher-capacity $r = 32$ setting is more prone to overfitting or unstable specialization, whereas $r = 16$ provides a better bias–variance trade-off: it is expressive enough to adapt to the Pokémon caption domain while remaining sufficiently constrained to generalize. Conversely, ranks smaller than $r = 16$ performed worse in earlier trials, consistent with underfitting due to insufficient adapter capacity. Therefore, based on empirical validation and qualitative inspection, we selected $r = 16$ and $\alpha = 32$ as the final LoRA configuration.

**Interpretation of fixing $\alpha$ while changing $r$.** Because the LoRA update is scaled by $\alpha/r$,

$$\Delta W = \frac{\alpha}{r} BA,$$

keeping $\alpha = 32$ while decreasing $r$ from 32 to 16 increases the scaling factor from 1 to 2. This yields stronger updates per learned direction while still using fewer trainable parameters overall. In our experiments, this combination improved adaptation without requiring the larger parameter count of $r = 32$, matching the low-data regime more effectively.

## 3.3 Training Dynamics and Loss

To evaluate the stability of our chosen architecture, we monitored the training and validation loss throughout the adaptation process.

As illustrated in Figure 2, the model exhibits a smooth, synchronized descent in both training and validation losses during the initial epochs. The validation loss stabilizes and achieves an optimal value of approximately 1.70 before plateauing. This trajectory highlights the parameter efficiency of the text-only model; by utilizing roughly half the trainable parameters of a dual-encoder setup, it achieves comparable performance while mitigating the severe overfitting often indicated by divergent training losses in more complex architectures. Consequently, utilizing the checkpoint with the lowest validation loss ensures the model captures domain-specific patterns without losing generalization.

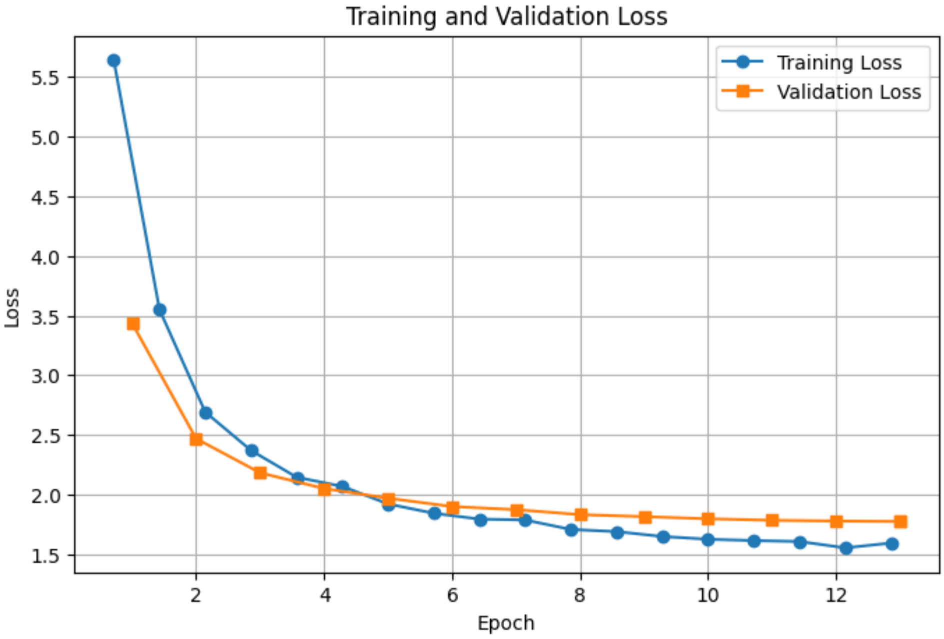


Figure 2: Training and validation loss across epochs for the optimal text-only LoRA configuration ($r = 16, \alpha = 32$).

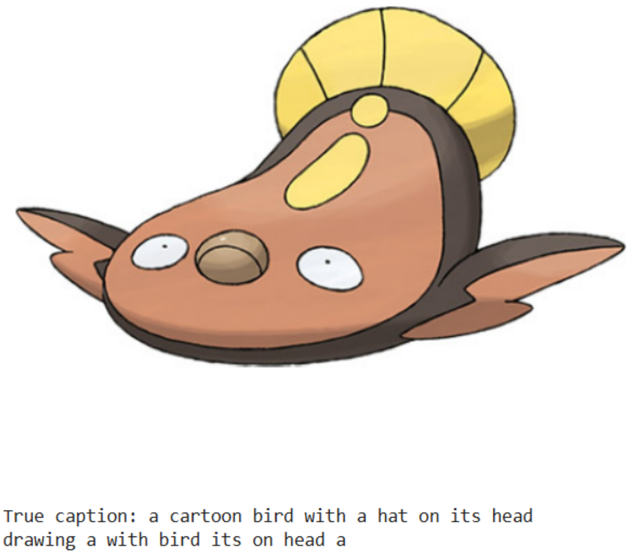
## 4 Inference and Generation Parameters

For qualitative inspection of the trained model, we generate captions using constrained beam search rather than stochastic sampling. In our experiments, beam search with a small beam width produced more stable and semantically correct captions on the Pokémon subset, while sampling occasionally introduced irrelevant or unstable phrasing.
Specifically, we use:

$$\texttt{num\_beams} = 2, \quad \texttt{max\_new\_tokens} = 40, \quad \texttt{length\_penalty} = 1.0,$$

together with repetition-control constraints `no_repeat_ngram_size` $= 2$ and `repetition_penalty` $=$ 1.15, and we enable `early_stopping=True`. A small beam (2) balances diversity and fluency: it is less prone to the high-probability function-word loops sometimes observed with larger beams, while still being more deterministic than sampling. Setting `length_penalty` $= 1.0$ avoids biasing the model toward overly short captions (values $< 1$) or overly long captions (values $> 1$). The repetition constraints reduce degenerate sequences (e.g., repeated articles or prepositions) and improve readability without overly restricting legitimate phrase reuse. Finally, early stopping terminates generation once the end-of-sequence condition is reached, preventing unnecessary continuation.
The final generation configuration used is shown below:

```
generated_ids = model.generate(
    pixel_values=inputs["pixel_values"],
    max_new_tokens=40,
    num_beams=2,
    length_penalty=1.0,
    no_repeat_ngram_size=2,
    repetition_penalty=1.15,
    early_stopping=True,
)
```



```
True caption: a cartoon bird with a hat on its head
drawing a with bird its on head a
```

Figure 3: An example from the dataset demonstrating an expected noisy ground-truth caption.

As shown in Figure 3, some ground-truth labels within this small corpus contain poorly structured or nonsensical text sequences (e.g., "drawing a with bird its on head a"). In such a low-data regime, encountering this level of noise is expected. Aggressively filtering or discarding these malformed samples was not feasible, as it would critically reduce the already scarce volume of training data available for fine-tuning. Consequently, the adapter must learn from these imperfect labels, which inherently places an upper bound on the syntactic coherence the model can achieve.

## 5 Conclusion

This study demonstrates that applying LoRA effectively stabilizes training on a constrained 700-image dataset, preventing the severe overfitting seen in full fine-tuning. Crucially, the ablation study reveals that adapting the text decoder alone is sufficient to bridge the domain gap. By further optimizing the Text-Only LoRA setup with an expanded rank ($r = 32$), we achieve the most parameter-efficient pipeline, resulting in a validation loss of 1.704 without wasting computational resources on vision-encoder memorization.

## 6 Future Work and Open Questions

While the Text-Only approach proved most efficient, the inability of the Vision adapters to improve validation scores raises an important architectural question regarding domain shift. The `git-base` vision encoder was pre-trained on realistic photographs, yet the Pokémon dataset consists entirely of 2D stylized illustrations.

Logically, updating the vision weights *should* have helped the model parse this new 2D art style. The failure to do so suggests that the 700-image dataset may simply be too small to successfully re-align the vision encoder's latent space without immediate overfitting. Future research should investigate whether increasing the dataset size, or applying a much higher LoRA dropout rate specifically to the vision modules, would unlock the theoretical benefits of Dual-Encoder adaptation.