Fine-Tuning a Vision-Language Model for Generative Visual Grounding

[Kazybek Askarbek]

June 19, 2025

Abstract

This report details the process of fine-tuning a state-of-the-art multimodal model, Google's PaliGemma-2, for the task of visual grounding. The objective is to enable the model to identify specific objects within an image and output their precise coordinates in a structured text format. We leverage the allenai/pixmo-points dataset and employ a parameter-efficient fine-tuning (PEFT) technique, specifically Quantized Low-Rank Adaptation (QLoRA), to train the 3-billion parameter model on a dual-GPU setup. The project demonstrates the viability of reformulating a classic computer vision problem as a conditional text generation task, achieving strong quantitative results based on coordinate-level regression metrics. The complete project code is available on GitHub:

Project Repository: https://github.com/QazyBi/point-vlm

1 Introduction

Visual grounding, the task of linking textual descriptions to specific regions in an image, is a cornerstone of multimodal artificial intelligence. It is a critical capability for applications ranging from robotics and human-computer interaction, where a machine must understand commands like "pick up the red cup," to advanced image retrieval and accessibility tools for the visually impaired.

Traditionally, this task has been approached through object detection or segmentation, where models output bounding boxes or pixel masks. While effective, these methods often require complex model architectures and specialized loss functions. This project explores a modern, alternative paradigm: treating visual grounding as a conditional language modeling problem. Our approach is unique in its combination of a state-of-the-art generative Vision-Language Model (VLM), PaliGemma-2, with a highly efficient fine-tuning strategy, QLoRA. Instead of predicting bounding boxes, we train the model to directly generate a string containing the normalized (x, y) coordinates of the target object. This method offers several advantages:

- **Simplicity:** It uses the standard cross-entropy loss function inherent to language models, eliminating the need for complex regression or IoU-based losses.
- **Flexibility:** The generative format can be easily extended to support more complex outputs, such as describing multiple points, shapes, or even reasoning about object locations.
- **Efficiency:** By using QLoRA, we can fine-tune a massive 3-billion parameter model with limited computational resources, making advanced VLM research more accessible.

This report will detail the methodology, from data preparation and model setup to a rigorous evaluation of the final system.

1.1 Team

Kazybek Askarbek was responsible for all aspects of the project, including literature review, data processing, model implementation and training, experimental design, and the preparation of this report.

2 Related Work

The problem of visual grounding has been explored through several distinct paradigms in computer vision and NLP.

Detection-Based Approaches: The most common approach frames visual grounding as a specialized object detection task. Early methods would generate a set of region proposals and then rank them based on their similarity to the input text query. Modern approaches, such as those inspired by DETR (DEtection TRansformer) (Carion et al., 2020), adopt an end-to-end philosophy. These models fuse visual features from a CNN or Vision Transformer backbone with text embeddings and use a transformer decoder to directly predict a set of bounding boxes and class labels. While powerful, they are fundamentally discriminative models trained to output fixed-format bounding boxes.

Segmentation-Based Approaches: A more fine-grained approach involves predicting a pixel-level mask for the object described in the text. This is often referred to as referring expression segmentation. Recent advancements, most notably the Segment Anything Model (SAM) (Kirillov et al., 2023), have introduced promptable segmentation. SAM can take points, boxes, or text as prompts to generate high-quality masks, demonstrating the power of multimodal inputs for precise localization. However, training such models from scratch is computationally intensive.

Generative and Coordinate-Based Approaches: Our work falls into an emerging category of generative approaches. Instead of predicting spatial coordinates through regression, these models are trained to output the coordinates as a sequence of text tokens. This approach was popularized by models

like Pix2Struct (Lee et al., 2023), which was designed for parsing visual information from screenshots and documents. The core idea is to discretize the continuous space of coordinates into a vocabulary of tokens (e.g., "0", "1", ..., "9", ".") and train the model to generate the correct sequence. The PaliGemma model itself, in its original training, was exposed to tasks involving coordinate generation, making it a suitable foundation for our fine-tuning.[1] Our project builds directly on this paradigm, applying it specifically to the open-domain visual grounding task defined by the pixmo-points dataset.

3 Model Description

Our approach centers on fine-tuning the google/paligemma2-3b-pt-224 model. This is a generative VLM that combines a powerful vision encoder with a capable language model, enabling it to process and generate interleaved image and text data.

3.1 Core Architecture

The PaliGemma-2 model consists of three main components:

1. **Vision Encoder:** A SigLIP-400m/32 vision transformer that processes an input image and converts it into a sequence of feature embeddings.
2. **Language Model:** A Gemma-2B decoder-only language model that processes textual information and generates the output sequence.
3. **Projector:** A multi-modal projector network that maps the vision embeddings into the same vector space as the text embeddings, allowing the language model to "understand" the image content.

3.2 Input and Output Formatting

A key aspect of our approach is reformulating the task. We instruct the model using a natural language prompt that includes a special ‘<image>‘ token, followed by an instruction. The model's expected output is a structured string containing the coordinates.

Input Prompt Format: ‘"<image>Point to label"‘

Target Output Format: ‘"<point x="x_{coord}"y = "y_{coord}" > $label$ < /$point$ > "‘

For multiple points, the format is: ‘"<points x1="x1" y1="y1" x2="x2" y2="y2"...>label</points>"‘

This strategy converts the regression problem of finding coordinates into a sequence-to-sequence text generation problem, which is the native task of the underlying language model.

3.3 Parameter-Efficient Fine-Tuning (PEFT) with QLoRA

To make training a 3-billion parameter model feasible, we employ Quantized Low-Rank Adaptation (QLoRA) (Dettmers et al., 2023). QLoRA combines two powerful techniques:

1. **4-bit Quantization:** The weights of the pre-trained model are quantized from their native 16-bit format (bfloat16) to an efficient 4-bit format (using the NF4 data type). This dramatically reduces the memory footprint of the model.

2. **Low-Rank Adaptation (LoRA):** The quantized, frozen base model is augmented with small, trainable "adapter" matrices. For a given linear layer with weight matrix W_0, the forward pass is modified as $Y = W_0X + BA(X)$, where A and B are the low-rank adapter matrices. Only A and B are updated during training.

In our implementation, we targeted all linear layers in the attention and feed-forward network blocks of the language model for LoRA adaptation, using a rank (‘r‘) of 8. This results in only ∼10.4 million trainable parameters, less than 0.4% of the total model size, while preserving near-full fine-tuning performance.

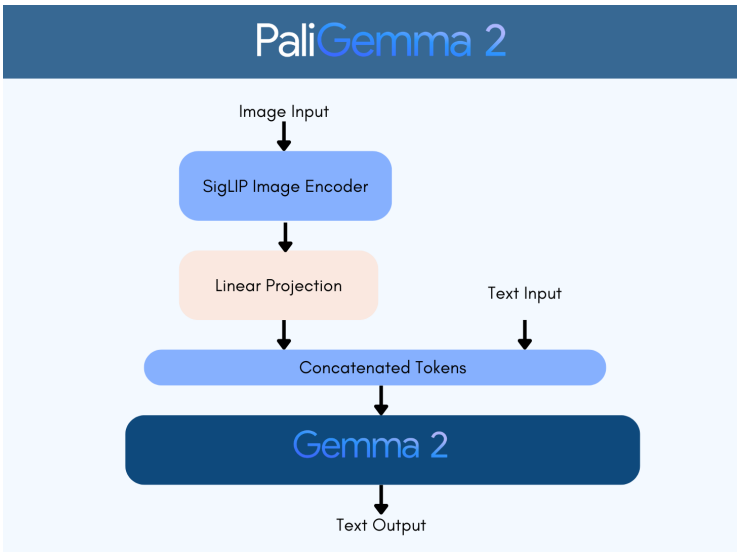

Figure 1: A high-level diagram of our approach. An image and a text prompt are fed into the PaliGemma-2 model. The model, fine-tuned with QLoRA, generates a structured string containing the object’s coordinates, which is then parsed for evaluation.

4 Dataset

This project utilizes the **PixMo-Points** dataset, available on the Hugging Face Hub as allenai/pixmo-points. This dataset was created as part of the work on PixMo (Sadeh et al., 2024), designed for training models on fine-grained visual localization tasks. The dataset can be accessed directly using the Hugging Face datasets library. The dataset consists of image-text pairs where the images are sourced from the large-scale LAION dataset. Each sample contains:

- An `image_url` pointing to the original image.
- A `label` describing an object in the image.
- A list of `points`, where each point is a dictionary containing the normalized `x` and `y` coordinates of an instance of the labeled object.

4.1 Data Preparation and Filtering

We began with a random subset of 100,000 samples from the training split. Our preparation pipeline involved several steps:

1. **Downloading and Caching:** All images were downloaded from their URLs and cached locally to accelerate access during training.
2. **Filtering:** Samples were filtered to ensure data quality. We removed any samples for which the image download failed, that did not contain any point annotations, or that corresponded to corrupted or invalid image files (e.g., non-RGB or 1-dimensional images).
3. **Image Resizing:** All images were resized to 224x224x3 to match the resolution of pretraining images.

After this filtering process, a cleaned and validated dataset was ready for training. The statistics of this process are shown in Table 1.

Stage	**Number of Samples**
Initial random subset	100,000
After downloading and caching	99,574
After filtering for non-empty points	71,328
After final image validation	70,914
Final Training Set (90%)	**63,822**
Final Validation Set (10%)	**7,092**

Table 1: Statistics of the dataset preparation and splitting process.

5 Experiments

5.1 Metrics

Evaluating a generative model that outputs coordinates requires a different approach from traditional object detection metrics like Intersection-over-Union (IoU). Our evaluation directly assesses the numerical accuracy of the generated points. This methodology is computationally efficient, as it avoids the need for resource-intensive post-processing steps like using an object segmentation model (e.g., SAM) to convert points into masks for IoU calculation.[2] We use the following regression and distance-based metrics, calculated between the set of predicted points and ground-truth points for each sample:

- **Mean Squared Error (MSE):** Calculated independently for the x and y coordinates (mean_mse_x, mean_mse_y). MSE penalizes larger errors more significantly, making it sensitive to outliers where the model's prediction is far from the ground truth.[3]
- **Mean Absolute Error (MAE):** Calculated for x and y coordinates (mean_mae_x, mean_mae_y). MAE provides a direct average magnitude of the error in pixels for each axis.
- **Distance Error:** The mean Euclidean distance between each ground-truth point and its closest predicted point. This is an intuitive single-value metric for localization accuracy.
- **Normalized Distance Error:** The Distance Error normalized by the image diagonal length. This allows for a fair comparison of errors across images of potentially different original sizes.
- **Success Rate:** A binary success metric. A prediction is considered successful if the distance error is less than a threshold of 5

Future Improvements: While these metrics effectively evaluate pointing accuracy, a future step could involve a more semantically rich evaluation. By feeding the model's predicted point into a powerful segmentation model like SAM, one could generate an object mask.[2] Comparing this mask to a ground-truth segmentation would allow for the calculation of an IoU, assessing not just if the point is correct, but if it correctly identifies the full extent of the intended object.

5.2 Experiment Setup

The model was fine-tuned for one epoch using the Hugging Face Trainer on a system with two NVIDIA H100 GPUs. The dataset was split into a 90% training set and a 10% validation set. Key hyperparameters are listed in Table 2.

Hyperparameter	Value
Base Model	‘google/paligemma2-3b-pt-224‘
Epochs	3
Batch Size per Device	4
Gradient Accumulation Steps	4
Effective Batch Size	16
Optimizer	`paged_adamw_8bit`
Learning Rate	1×10^{-4}
Precision	bfloat16
QLoRA Configuration	
LoRA Rank (r)	8
LoRA Alpha	8
Target Modules	`q/k/v/o_proj`, `gate/up/down_proj`

Table 2: Key hyperparameters used for fine-tuning.

6 Results

Our fine-tuned model demonstrates a strong capability to generate accurate spatial coordinates from textual prompts. The quantitative results, evaluated on a hold-out set of 100 validation samples, are summarized in Table 3. The

Metric	Score
Number of Valid Predictions	79 / 100 (79%)
Mean Distance Error	22.31
Mean Normalized Distance Error	0.0211
Median Distance Error	12.88
Standard Deviation of Distance Error	19.33
Mean MSE (X / Y)	658.67 / 303.09
Mean MAE (X / Y)	16.73 / 10.73
Mean Success Rate	**0.9011**

Table 3: Quantitative results on a 100-sample validation set. The Success Rate is calculated only on the 79 valid predictions.

model achieves a high **Mean Success Rate of 90.1%**, indicating that when it produces a valid output, the predicted point is located very close to the ground truth. The mean_normalized_distance_error of just 2.1% of the image diagonal further underscores the model's high precision. However, a key challenge identified during evaluation is the reliability of the structured output generation. The model produced syntactically correct and parsable coordinates for only 79 of the 100 test samples, resulting in a 21% failure rate on this set. These failures, logged as "Could not parse model output," typically occurred when the model generated incomplete <points> tags, as shown in Table 4. This

suggests that while the core visual grounding capability is strong, the model's ability to adhere to the rigid output format needs further improvement.[4][5] The discrepancy between the mean distance error (22.31) and the much lower median distance error (12.88), combined with a high standard deviation (19.33), suggests that the average error is influenced by a subset of predictions with large errors. This is also reflected in the high MSE values, which penalize these outliers more heavily than MAE.[3] Qualitative analysis supports these findings. As seen in Figure 2, the model can accurately locate common objects. Table 4 highlights a successful case, a case with a large distance error (pointing to the wrong instance of a "boat"), and a structural failure case.

Figure 2: A qualitative example from the validation set. The ground-truth point is shown as a blue circle, and the model's predicted point is the red 'X'. The model successfully locates the "left hand" with a low distance error.

7 Conclusion

In this project, we successfully fine-tuned the PaliGemma-2 vision-language model for a generative visual grounding task. By leveraging the parameter-efficient QLoRA technique, we demonstrated that it is feasible to adapt a large-scale VLM for this specialized task using limited computational resources. Our model achieved a high degree of localization accuracy, evidenced by a **Mean Success Rate of 90.1%** and a Mean Normalized Distance Error of just 2.1% on validly generated outputs. The results confirm the efficacy of formulating

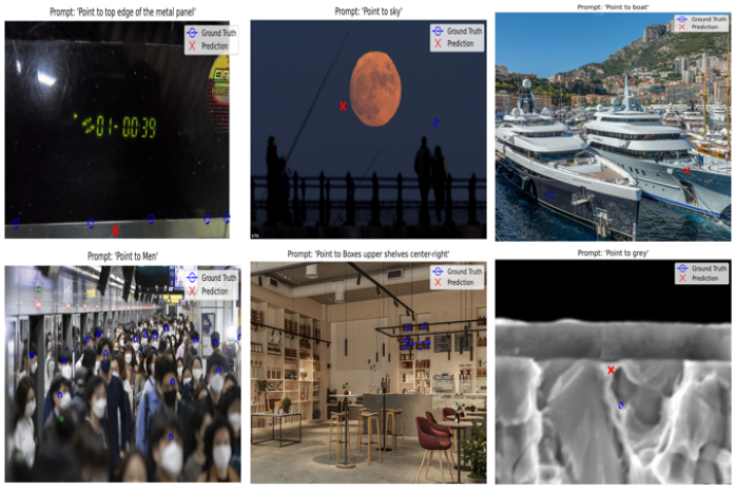

Figure 3: Multiple examples from the validation set showing predictions (red 'X') and ground truth (blue circle). The model demonstrates variability in its accuracy.

Successful Case (Label: "sky")
Model Output: <point x="39.04" y="39.16">sky</point>
Failure Case (Large Error) (Label: "boat")
Model Output: <point x="81.13" y="69.04">boat</point>
Failure Case (Parsing Error) (Label: "men")
Model Output: <points x1="19.0" y1="50.0" x2="31.0" y2="50.0" ... x7

Table 4: Sample model outputs from the validation set illustrating a successful prediction, a prediction with a large localization error, and a syntactically incomplete output that could not be parsed.

coordinate prediction as a conditional text generation problem. However, the evaluation also highlighted a significant challenge in the model's reliability to produce syntactically correct structured output, with a 21% parsing failure rate on the test set. Future work should focus on two main areas: first, improving the robustness of the structured output generation, perhaps through more advanced prompting techniques or constrained decoding methods.[5] Second, enhancing instance discrimination to reduce large errors in scenes with multiple similar objects. Overall, this project serves as a strong proof-of-concept for the power and flexibility of modern generative VLMs, while also clearly identifying key areas for future research and improvement.

References

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. *European conference on computer vision*, pages 213–229.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. (2023). Qlora: Efficient finetuning of quantized llms.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. (2023). Segment anything. *arXiv preprint arXiv:2304.02643*.

Lee, K., Ainslie, W., Driga, M., E-K, F., Rappoport, N., Savu, A., Tsvyashchenko, S., Xu, H., Z-G, T., and Houlsby, N. (2023). Pix2struct: Screenshot parsing as pretext for vqa. *International Conference on Machine Learning*, pages 18971–18991.

Sadeh, N. et al. (2024). Pixmo: A cross-modal generalist model for vision, language, and more.