Quantization of Vision-Language Model for Remote Sensing Visual Grounding

Shamma Alblooshi 100045387

Suhail Almarzouqi 100045490

Tasneim Aldhanhani 100048765

Course: Vision Language Models – COS 794

Date: May 7, 2025

1 Description of the Research Problem

The increasing availability and resolution of remote sensing (RS) imagery have necessitated the development of efficient processing and retrieval methodologies. The integration of vision-language models (VLMs) with RS imagery has facilitated advancements in tasks such as image captioning, image-text retrieval, and visual question answering. However, remote sensing visual grounding (RSVG), the task of localizing objects in RS images based on natural language queries remains underexplored. RSVG plays a crucial role in applications such as military target detection, disaster monitoring, precision agriculture, urban planning, and search and rescue operations. Despite recent efforts to enhance RSVG through transformer-based models [1], challenges persist in terms of computational efficiency, scalability, and real-time deployment.

VLMs have demonstrated state-of-the-art performance in multimodal learning but remain computationally expensive, limiting their feasibility for deployment in resource constrained environments such as satellites, drones, and edge devices. To address this limitation, quantization techniques have emerged as a promising solution, reducing model size and inference latency while preserving performance. However, a key challenge lies in determining how post-training quantization (PTQ) can be effectively applied to an existing VLM specialized in visual grounding tasks for remote sensing data while maintaining accuracy. Additionally, it is critical to assess the trade-offs between the computational efficiency gains from quantization and the potential degradation in model accuracy for visual grounding tasks in remote sensing imagery.

This project aims to apply post-training quantization strategies on a pre-trained vision-language model for the specific task of remote sensing visual grounding. We aim to answer the following two questions.

- How can post-training quantization be applied to an existing VLM specialized in visual grounding tasks for remote sensing data?
- What is the trade-off between the gains from quantization and the potential decrease in accuracy for visual grounding in satellite imagery?

2 Related Work and literature

Over the past decade, visual grounding has evolved significantly from traditional region-based detection pipelines to more sophisticated multimodal and generative frameworks. Visual grounding enables flexible, open-world localization of objects or regions in an image based on natural language expressions, moving beyond static, closed-set object categories toward more generalized spatial reasoning. The task has been addressed under various learning settings and architectural designs. Within the fully supervised paradigm, Xiao et al. [2] categorize methods into five primary technical routes: traditional CNN-based models, Transformer-based architectures, visual-language pretraining (VLP) models, grounding-oriented pretraining strategies, and grounding multimodal large language models (GMLLMs). These developments reflect a shift from two-stage proposal-based systems to end-to-end Transformer-based and instruction-following models capable of handling tasks such as referring expression comprehension, segmentation, captioning, and multi-turn dialogue.

Beyond supervision level, visual grounding methods are also distinguished by their learning paradigms. These include fully supervised, weakly supervised, semi-supervised, unsupervised, zero-shot, multi-task, and generalized visual grounding (GVG) [2]. Fully supervised approaches rely on annotated image-query-region triplets, while weakly and semi-supervised models leverage unlabeled or partially labeled data through strategies like pseudo-labeling and cross-modal alignment. Unsupervised methods forgo spatial annotations entirely, learning instead from visual-textual co-occurrence. Zero-shot grounding uses pretrained vision-language models to generalize to unseen concepts without fine-tuning. Multi-task and GVG frameworks enable a single model to handle diverse or ambiguous queries, further pushing toward scalable, real-world applicability.

Detection-based grounding initially used proposal scoring based on multimodal similarity, but recent methods such as MDETR and TransVG apply cross-modal attention to directly align visual and textual inputs. Segmentation-based methods, including referring expression segmentation (RES), provide fine-grained pixel-level localization, useful for dense or overlapping scenes. More recently, generation-based grounding models particularly GMLLMs like LLaVA and Ferret have unified grounding with natural language generation through instruction tuning and autoregressive modeling. These models produce spatially grounded captions and multi-turn conversational responses, marking a significant step toward general-purpose visual grounding systems.

Recent advancements in RSVG have introduced transformer-based models such as RSVG [1] and GeoGround [3], which improve vision-language alignment through multi-level feature extraction and task-specific learning strategies. RSVGD utilizes a Multi-Level Cross-Modal (MLCM) feature learning module, which enhances object localization in cluttered RS scenes using multi-scale visual features and textual embeddings. GeoGround, on the other hand, employs text-mask techniques and geometry-guided learning, supporting multiple localization formats, including horizontal bounding boxes (HBB), oriented bounding boxes (OBB), and segmentation-based localization. GeoChat [4] has been introduced as a large-scale, instruction-following vision-language model specifically adapted for remote sensing. By leveraging a CLIP-based vision encoder and Vicuna LLM backbone, GeoChat aligns visual and textual modalities through instruction tuning and generates natural language responses across a variety of grounding tasks. It further demonstrates robust zero-shot generalization across region captioning, referring expression comprehension, and scene-level VQA tasks. While these models achieve high accuracy, their computational requirements hinder real-time deployment.

To mitigate these constraints, this report explores post training quantization techniques as a means of optimizing RSVG models for efficient deployment. Quantization reduces computational complexity by representing model weights and activations in lower-bit precision formats (e.g., 8-bit integers instead of 32-bit floating points), significantly decreasing memory footprint and inference latency [5]. However, quantization may introduce accuracy degradation, necessitating optimization techniques such as Post-Training Quantization (PTQ) and Quantization-Aware Training (QAT) to balance efficiency and performance [6, 5]. An overview of various quantization techniques is provided in the Appendix for reference.

3 Methodology

In this work, we propose a post-training quantization pipeline for the remote sensing visual grounding model, GeoChat [4]. Our approach follows three main steps. Firstly, We use the GeoChat

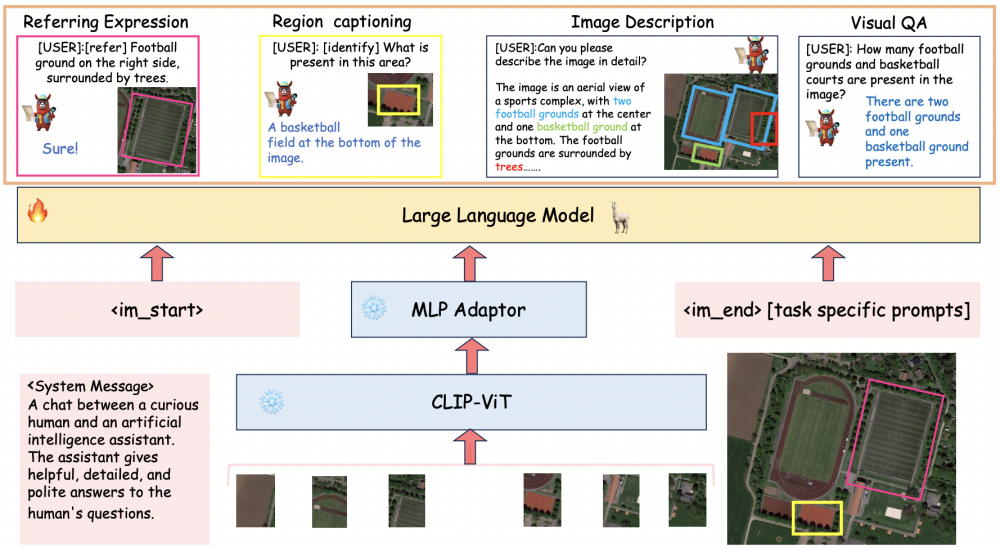

Figure 1: GeoChat Model Architecture [4]

model in BF16 (Brain Float 16) precision as a baseline for comparison. Next, we apply 8-bit quantization using the Bits and Bytes library [7]. Finally, we further reduce precision to 4-bit using the custom QVLM model [8], a variant built upon Bits and Bytes quantization techniques. In the sections below, we detail each step.

3.1 GeoChat Model

GeoChat was developed to address the limitations of general-domain vision-language models (VLMs) when applied to remote sensing imagery. Standard VLMs are typically trained on natural image datasets and therefore struggle with remote sensing images that are characterized by high resolutions, small objects, and significant scale variations. These challenges necessitate specialized region-level reasoning and spatial grounding to accurately interpret remote sensing data. GeoChat fills this gap by supporting versatile, multitask conversations on remote sensing images and by providing spatially grounded responses, thus acting as a robust baseline model for applications such as quantization in remote sensing contexts.

Its architecture enables it to operate effectively at different levels of granularity. The first one is image-level conversation tasks where the model processes an image x along with a user query q without additional spatial cues. In this mode, GeoChat tackles holistic tasks such as visual question answering, scene classification, and image captioning by considering the entire scene. The second one is the region-level conversation tasks where it incorporates spatial box locations b with the image and query. Allowing GeoChat to focus on specific regions. This enables it to perform region captioning, answer region-specific questions, and support multi-turn dialogues centered on particular areas of the image. The last one which is the grounded conversation task, it uses special task-specification tokens t to generate descriptive responses and also output precise object locations.

3.1.1 Geochat Architecture

GeoChat builds upon the architecture of LLaVA-v1.5, which includes three main components: i) Global Image encoder, ii) an MLP adaptor (two linear layers) and iii) LLM. However, different to LLaVA, they add a specific task prompt that indicates the type of task desired from the model. The architecture is shown in figure 1. We describe each component in the architecture as follows:

Dual-Stage Visual Backbone: The geochat arhitecture uses the pretrained CLIP-ViT(L-14) as the visual encoder, initially designed for natural images. The model enhances resolution by interpolating the original positional encodings to support larger input images (e.g., 504×504). This change increases the effective number of image patches (from 576 to 1296), which is essential for processing high-resolution details found in remote sensing imagery.

Modular MLP Adaptor: A two-layer MLP adaptor is used to project visual features from the frozen CLIP-ViT into the language model space. This adaptor allows the model to map 1024-dimensional vision tokens into a 4096-dimensional space, matching the input requirements of the underlying large language model (LLM).

Large Language Model Integration: The architecture leverages the open-source Vicuna-v1.5 (7B) LLM to generate responses and understand instructions. Importantly, by using LoRA (Low-Rank Adaptation), GeoChat achieves efficient fine-tuning. LoRA updates a pair of small matrices instead of the full weight matrices, which speeds up training and helps retain the general instruction-following abilities of the base LLM.

Task-Specific Tokens and Spatial Representation: GeoChat introduces distinct task tokens (e.g., [grounding], [identify], [refer]) to seamlessly switch between tasks each for grounded conversations, region captioning and referring expression comprehension. As for the case of visual question answering and scene classification, they directly ask the model to output the answer in a single word or phrase The model represents bounding boxes using a textual format: $b = bxleft, bytop, bxright, bybottom|\theta$. This format (with normalized coordinates and a rotation angle) allows for precise spatial reasoning and is central to grounding outputs in remote sensing imagery.

3.1.2 Datasets

To compensate for the lack of remote sensing specific multimodal data, the authors constructed a large scale instruction following dataset comprising nearly 318,000 image instruction pairs. The dataset is formed by aggregating: object detection datasets like DOTA, DIOR, and FAIR1M form the basis for region-level tasks by providing bounding box annotations. In addition to adding scene classification and VQA datasets: NWPU-RESISC-45 (scene classification) and LR-BEN (visual question answering), along with Floodnet (for flood detection), are integrated to extend the model's ability to handle diverse task types.

GeoChat was extensively evaluated across several key remote sensing tasks. For scene classification, the model was evaluated on datasets such as AID and UCMerced. Wheres for visual question answering (VQA) it was evaluated on RSVQA-LRBEN and RSVQA-HRBEN datasets.

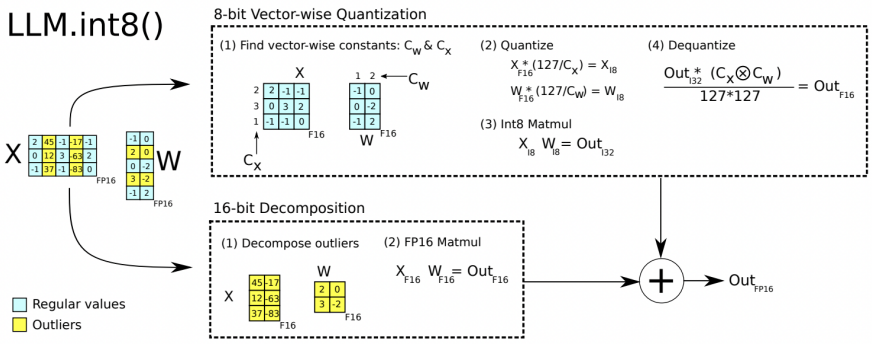

Figure 2: LLM.int8() method [7]

3.2 8-bit Quantization via Bits and Bytes

Post-training quantization was first applied to convert the baseline BF16 model to 8-bit precision. We leveraged the Bits and Bytes library, which implements an asymmetric linear quantization scheme to map the full-precision weights into 8-bit integers.

The Bits and Bytes Library includes LLM.INT8() [7], which is a quantization method that aims to make LLMs more accessible without significant degradation. This is different than the naive 8 bit quantization which can result in loss of critical information and accuracy. This method dynamically adapts to ensure sensitive components of the computation retain higher precision when needed. The workflow of the 8-bit quantization is shown in Figure 2.

LLM.int8() accelerates and compresses Transformer matrix multiplications by splitting the computation into two parallel paths, a low-precision 8-bit path for the vast majority of "regular" activations and weights, and a full FP16 path handling the sparse, high-magnitude outliers then recombines them to exactly recover the FP16 result.

1. **Vector-Wise 8-Bit Path**
   - *Per-Vector Scaling.* Compute an independent normalization constant for each row of the activation matrix $X \in \mathbb{R}^{s\times h}$, denoted $C_x \in \mathbb{R}^s$, and for each column of the weight matrix $W \in \mathbb{R}^{h\times o}$, denoted $C_w \in \mathbb{R}^o$.
   - *Quantization.* Each entry is quantized by producing two 8-bit integer matrices $X^{(8)}$ and $W^{(8)}$.
   - *Integer MatMul & Dequantization.* Perform the fast int8 GEMM to rescale back to FP16.
2. **16-Bit Path**
   - *Outlier Extraction.* Identify the small set of feature dimensions whose magnitudes exceed a threshold (e.g. $|x| > 6$). Denote their indices by O.
   - *FP16 MatMul.* Extract submatrices and compute the output.
3. **Recombination.** Finally, sum the two partial results in FP16:

$$O^{(\mathrm{FP16})} = \underbrace{O_O^{(\mathrm{FP16})}}_{\text{outliers}} + \underbrace{\frac{1}{127^2}(C_x \otimes C_w) \odot \left(X^{(8)}W^{(8)}\right)}_{\text{normal features}}.$$

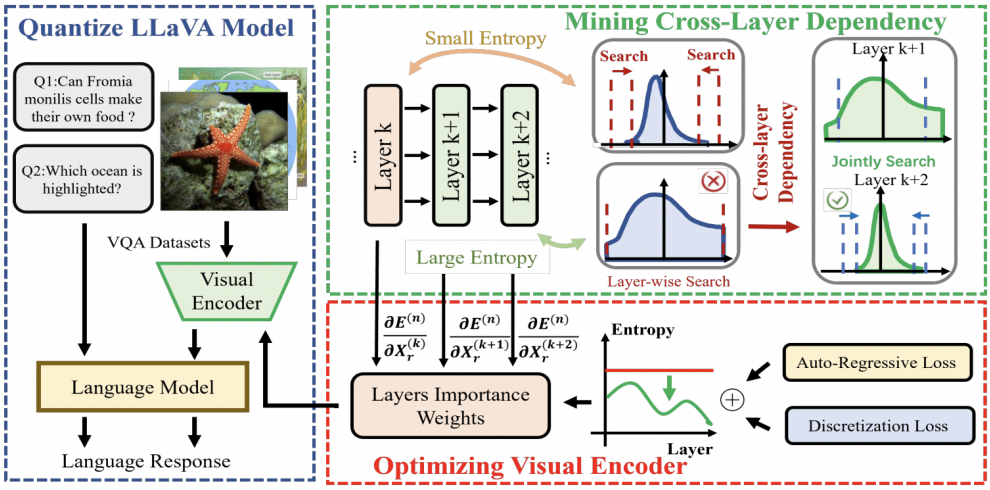

Figure 3: QVLM pipeline [8]

This procedure achieves nearly a 2× memory reduction compared to pure FP16 while preserving exact full-precision inference quality, even on models up to 175 B parameters.

3.3 4-bit Quantization via QVLM

Q-VLM [8] proposes a novel *post-training* 4-bit quantization framework that preserves multimodal reasoning accuracy without any additional fine-tuning. Conventional PTQ methods suffer from suboptimal rounding decisions when each layer is quantized in isolation, as errors accumulate through the network. The overall pipeline is shown in Figure 3.

The pipeline starts from a frozen LLaVA vision–language model that consists of a CLIP-based visual encoder feeding into a large language model (Vicuna). Given a small set of example VQA prompts and images (e.g. "Can Fromia monilis cells make their own food?"), they wish to replace every weight and activation tensor with a 4-bit integer representation, yet still produce the same open-ended textual responses. Rather than quantize each layer in isolation (which accumulates rounding error), they analyze how quantization noise propagates between layers.

In the top-right box they illustrate their discovery that whenever a layer's activation distribution has high entropy, its rounding error significantly affects downstream layers. Conversely, low-entropy layers can be re-quantized independently without harming later layers. Hence, they compute the *conditional entropy* between each pair of adjacent layers as a proxy for how strongly their quantization errors couple. Wherever this inter-layer dependency exceeds a threshold, they "lock" those consecutive layers into a single block and jointly search for the 4-bit clipping ranges that minimize that block's output error. Blocks with weaker dependencies can still be quantized one layer at a time.

Finally, the bottom-right box shows how, before quantization, they perform a brief calibration of the visual encoder itself. They attach two auxiliary losses. Discretization loss that encourages its activations to lie in narrow, low–dynamic-range bands (making them easier to quantize). An "entropy loss" weighted by each layer's influence on the final output error (measured via the Jacobian of block output error wrt that layer's activations). By back-propagating these objectives alongside the original auto-regressive language loss, the encoder learns to produce features whose

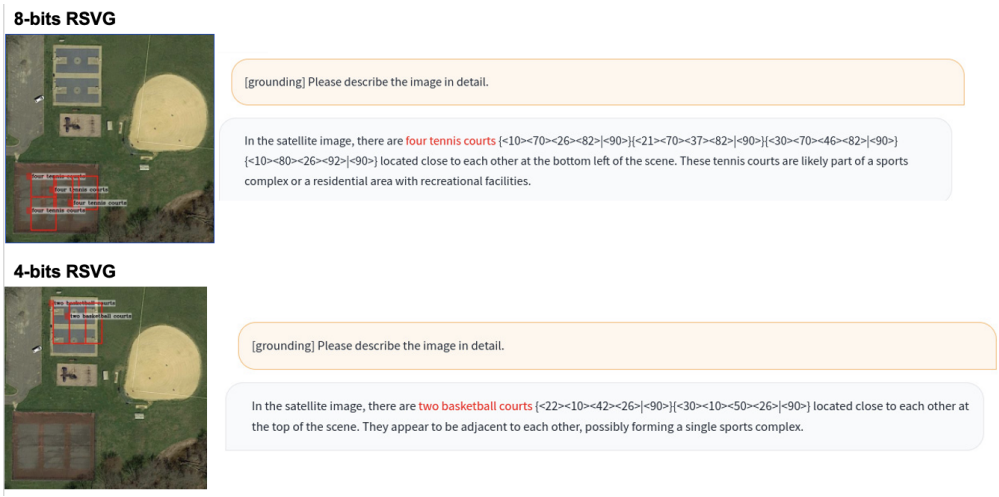

Figure 4: Visual Grounding Results

quantization errors no longer cascade strongly through the network.

4 Results and Discussion

4.1 Visual Grounding

Quantization had a positive impact on the visual grounding task. Table 1 shows the overall results compared to the baseline model. Both 8-bit and 4-bit quantized models maintained comparable visual grounding BF16 baseline, achieving a negligible drop in accuracy at IoU@0.5 and IoU@0.25 thresholds. This robustness in small loss of accuracy can be attributed to the regularization effect introduced by reduced numerical precision.

Post-training quantization suppresses minor overfitting to noisy visual and textual features common in complex remote sensing imagery, enhancing the model's generalization ability. Additionally, in remote sensing datasets with ambiguous object boundaries and cluttered backgrounds, slight smoothing caused by quantization helps stabilize grounding predictions, leading to more consistent object localization. Accuracy at 0.25 IoU (acc@0.25) applies a relaxed overlap threshold compared to acc@0.5, allowing more predictions to qualify as correct. This lower threshold is particularly useful for evaluating grounding performance on small or cluttered objects prevalent in remote sensing imagery.

Examples of the task are shown in the Figures 4 and 5. By introducing a light regularization effect, both 8-bit and 4-bit models stabilized their outputs and performed comparably to the BF16 baseline. Counting accuracy remained largely intact both quantized versions correctly counted the two Boeing 747s and the basketball courts, although the 8-bit model did overcount tennis courts and only minor scene interpretation differences arose under identical queries. Spatial grounding stayed coherent despite slight bounding-box "looseness" in the 4-bit model, demonstrating that even aggressive precision reduction can yield a compact, fast model without sacrificing descriptive fidelity for large, distinct targets.

We note, however, that the authors report these visual grounding results as being measured

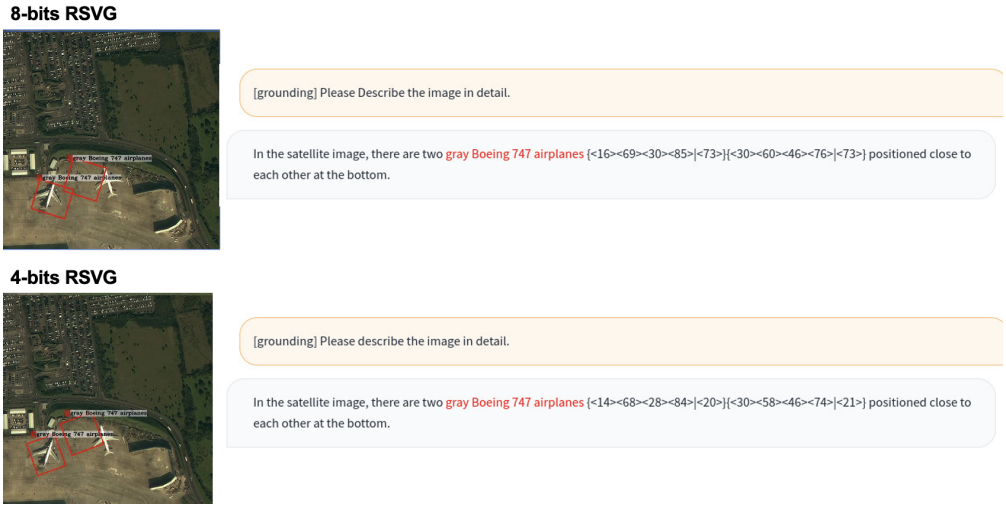

Figure 5: Visual Grounding Results

on the validation set, yet the evaluation code processes the entire dataset. Even if one restricts evaluation to the approximately 200 validation examples (out of 7,500 total), this represents only a small fraction of the full dataset, raising the possibility of misannotation or misreporting. Indeed, whereas the authors claim their BF16 baseline achieves 11.7% acc@0.5 and 33.9% acc@0.25, our replication on yields substantially higher figures of 18.1% and 42.4%, respectively which are reported on the table. This discrepancy suggests that the validation annotations are incomplete or the originally reported metrics are off.

Model	**acc@0.5**	**acc@0.25**
BF16 Model[1]	18.1	42.4
Quantized 8-bits	17.5	41.9
Quantized 4-bits	17.9	41

Table 1: Results on grounding description task.

4.2 Zero-Shot Scene Classification

Quantized models demonstrated strong robustness in zero-shot scene classification tasks. The 8-bit and 4-bit models maintained comparable classification accuracy to the BF16 baseline on the UCMerced and AID datasets, with negligible performance drops (less than 0.5%) shown in Table 2. Figure 6 shows that while the 8-bit model correctly distinguished between urban and rural scenes, the 4-bit model occasionally misclassified scenes under ambiguous visual conditions, reflecting slight sensitivity to extreme compression.

Overall, quantization had minimal impact on scene classification accuracy, confirming that lightweight models can retain strong scene understanding capabilities even under aggressive compression settings.

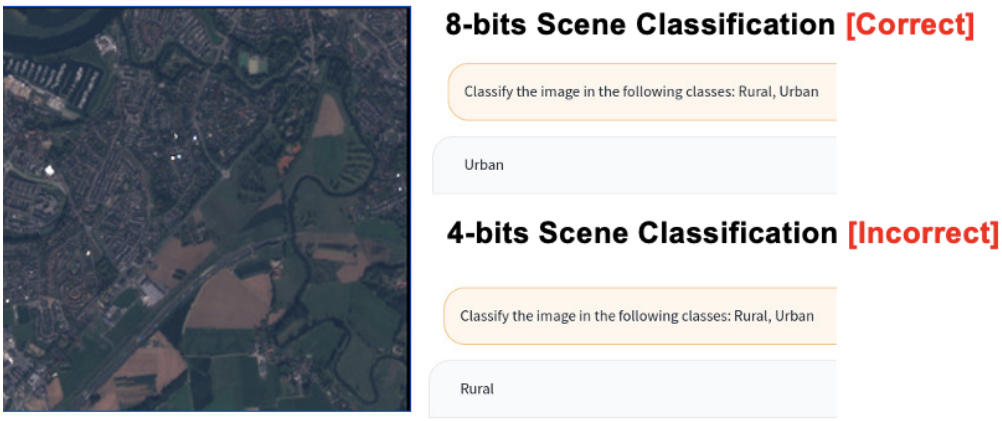

Figure 6: Zero-Shot Scene Classification Results

Model	UCMerced	AID
BF16 Model	84.43	72.03
Quantized 8-bits	84.00	72.1
Quantized 4-bits	83.95	71.70

Table 2: Zero-shot scene classification accuracy comparison on AID [9] and UCMerced [10] datasets.

4.3 Zero-Shot Remote Sensing Visual Question Answering (RSVQA)

In zero-shot remote sensing visual question answering (RSVQA), both the 8-bit and 4-bit models achieved similar or slightly lower average accuracies compared to the BF16 baseline as shown in Table 3. As shown in Figure 7, minor degradations were observed in detailed reasoning tasks, where the 4-bit model sometimes produced partial answers, missing specific entity names such as "Boeing 747." Nonetheless, the general ability to correctly answer scene-related questions and perform basic comparison and presence detection tasks was preserved across quantization levels.

Thus, despite slight descriptive losses, the quantized models demonstrated strong zero-shot generalization ability in RSVQA, supporting efficient deployment for real-world remote sensing applications.

Method	Presence	Comparison	Rural/Urban	Avg. Accuracy
BF16 Model	91.09	90.33	94.00	91.80
Quantized 8-bits	90.86	90.00	93.00	91.20
Quantized 4-bits	91.03	90.02	92.00	91.18

Table 3: General Zero-Shot on RSVQA-LRBEN [11] dataset for VQA task.

Figure 7: Zero-Shot RSVQA Results

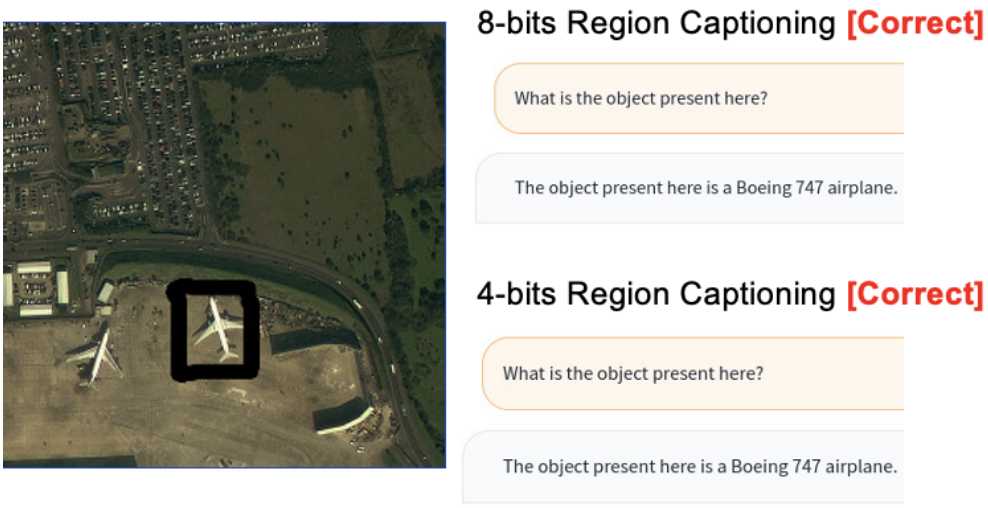

Figure 8: Region Captioning Results

4.4 Region-Level Captioning

In the region-level captioning task, both 8-bit and 4-bit models correctly generated descriptions identifying objects such as Boeing 747 airplanes, even after aggressive quantization. This is depicted in Figure 8. However, evaluation metrics (ROUGE-1, ROUGE-L, and METEOR) indicated a drop in textual fluency and lexical match scores compared to the BF16 baseline shown in Table 4. ROUGE-1 and ROUGE-L measure the overlap of individual words and sequences between generated and reference captions, emphasizing lexical precision, while METEOR provides a semantically informed evaluation by accounting for synonyms, word stems, and word order penalties.

The results indicate that while quantization moderately degrades language quality metrics, the essential semantic content and object grounding capabilities are preserved. Specifically, ROUGE-1 scores decreased by approximately 15.4%, ROUGE-L by about 15.8%, and METEOR scores declined by roughly 25.4% compared to the BF16 baseline. Despite this reduction in language quality scores, the main semantic content, specifically the correct object identity, was preserved across quantization levels.

Model	ROUGE-1	ROUGE-L	METEOR
BF16 Model	87.3	87.2	83.9
Quantized 8-bits	73.8	73.8	63.1
Quantized 4-bits	73.3	73.3	62.6

Table 4: Region level captioning performance.

5 Conclusion and Future Work

This work presented a comprehensive study on quantizing vision-language models (VLMs) for remote sensing visual grounding tasks. Through extensive evaluation, it was demonstrated that aggressive quantization to 8-bit and 4-bit precision significantly reduces model size and computational demands while preserving core grounding, scene classification, and question answering capabilities. Visual grounding performance was comparable to the BF16 baseline, which may be attributed to a light regularization effect introduced by reduced numerical precision. Zero-shot scene classification and RSVQA tasks remained robust under quantization, confirming the generalization strength of the compressed models.

However, analysis of region-level captioning revealed moderate degradations in language fluency and semantic richness, as reflected by decreases in ROUGE and METEOR scores. These observations highlight that while quantization supports efficient deployment of VLMs for real-time or near-real-time remote sensing applications, further refinements are required to preserve detailed reasoning and generation quality.

Future work will focus on enhancing model efficiency through advanced compression techniques. In addition to investigating mixed-precision quantization and group-wise quantization strategies to further reduce model size while minimizing performance degradation, structured and unstructured pruning methods will be explored to eliminate redundant weights and optimize computational cost. To address the observed degradation in fine-grained reasoning, approaches such as selective fine-tuning and knowledge distillation will be considered to improve answer completeness after quantization and pruning. Methods for improving spatial grounding precision, including calibration strategies and post-processing techniques, will also be developed to mitigate minor bounding box shifts introduced by aggressive compression. Furthermore, the scope of model optimization will be extended to additional remote sensing tasks such as change detection, damage assessment, and multi-modal retrieval. Future evaluations will benchmark performance on larger and more diverse datasets to assess model robustness under varying geographic and environmental conditions.

A Appendix

A.1 Quantization Techniques

Deep learning models, particularly large-scale architectures such as transformers and convolutional neural networks (CNNs), require substantial computational resources for both training and inference. Deploying these models on resource-constrained environments, such as edge devices, necessitates efficient model compression techniques. Quantization has emerged as a key optimization method that reduces the precision of model weights and activations, typically converting 32-bit floating-point (FP32) values into low-bit integer representations (e.g., INT8, INT4, or even 1-bit). This reduction in precision leads to lower memory usage, reduced computational complexity, and improved inference speed due to the efficiency of integer arithmetic [5, 6].

Quantization techniques can be broadly categorized based on calibration into Post-Training Quantization (PTQ) and Quantization-Aware Training (QAT) [5]. PTQ is widely adopted due to its simplicity, as it applies quantization to a trained model without retraining. It can be further divided into dynamic quantization, static quantization, GPTQ and GGUF. QAT, on the other hand, incorporates quantization during the training process, enabling the model to adapt to lower-bit representations and reducing accuracy loss after quantization. BitNet, a novel 1-bit quantization approach, falls under QAT, as it modifies the Transformer architecture itself to efficiently handle extreme quantization[6]. Additionally, dequantization techniques restore quantized values back to high-precision formats when needed for specific computations. This section provides an in-depth description of PTQ and QAT, highlighting their advantages, trade-offs, and cutting-edge advancements such as GPTQ, GGUF, and BitNet.

A.2 Post-Training Quantization (PTQ)

PTQ is a widely adopted method that applies quantization to a fully trained model without requiring retraining [5]. By reducing model precision, PTQ decreases memory footprint and computational costs, making deep learning models more efficient for deployment. PTQ can be classified into dynamic quantization, static quantization, and advanced low-bit quantization techniques such as GPTQ and GGUF.

A.2.1 Dynamic Quantization

Dynamic quantization is a post-training optimization technique applied during inference, ensuring that the training process remains unaffected. Prior to inference, model weights are converted to lower bit for example: INT8, while activations undergo dynamic quantization at runtime. This method enables optimized integer-based matrix multiplications, making it particularly effective for transformer-based models. Since activations are quantized on the fly, this approach introduces minimal overhead and maintains model accuracy. However, frequent integer-to-floating-point conversions can create a computational bottleneck, particularly when activations are repeatedly written and read in FP32 [12].

A.2.2 Static Quantization

Unlike dynamic quantization, static quantization precomputes and stores quantization parameters for both weights and activations before inference. This requires a calibration step, where representative input data is used to determine optimal scaling factors for converting values to lower precision. The precomputed quantization scheme eliminates the need for runtime conversions, improving inference speed. However, static quantization introduces an additional preprocessing step and may cause accuracy degradation due to discrepancies between training and inference precision [6].

A.2.3 GPTQ & GGUF

Quantization below 8-bit introduces higher quantization errors, making accuracy retention a challenge. However, 4-bit, 6-bit, and even 2-bit quantization have been developed to maximize compression while minimizing performance loss. The two most widely adopted 4-bit PTQ techniques are GPTQ (Generalized Post-Training Quantization) and GGUF (Generalized GPU-CPU Unified Format) [6]. GPTQ is an advanced quantization technique designed to optimize inference speed while minimizing accuracy loss. Unlike traditional post-training quantization methods that apply uniform quantization across layers, GPTQ employs asymmetric quantization and processes each layer independently. A key innovation of GPTQ is its use of the inverse-Hessian matrix, which quantifies weight sensitivity and redistributes quantization errors accordingly. This method ensures that critical weights retain higher precision, leading to minimal degradation in model performance. GPTQ is particularly well-suited for full-model GPU execution, leveraging optimized integer arithmetic for efficient computation [6].

GGUF is a hybrid quantization strategy that enhances model deployment flexibility by allowing layers to be dynamically offloaded between GPU and CPU. Unlike GPTQ, which is optimized for full-GPU execution, GGUF introduces block-wise quantization, where model weights are partitioned into super-blocks and sub-blocks, each with distinct scaling factors. This hierarchical quantization method enables precision retention while optimizing computation. By offloading select layers to the CPU, GGUF significantly reduces GPU memory constraints, making it particularly useful for devices with limited VRAM.

A.3 Quantization-Aware Training (QAT)

QAT differs from PTQ by integrating quantization effects directly into the training process [5]. Instead of applying quantization after training, QAT simulates low-bit behavior during training, allowing the model to adapt to lower precision representations. This technique reduces post-training accuracy loss compared to PTQ, requires additional computational resources during training, and is particularly useful for transformer-based NLP models and vision models, where accuracy is critical.

A.3.1 BitNet

One of the most advanced QAT-based quantization methods is BitNet, which introduces 1-bit quantization by modifying the Transformer architecture itself [6]. While traditional quantization techniques focus on reducing precision without altering model structures, BitNet takes a fundamentally

different approach by reengineering the underlying architecture to operate efficiently with extreme quantization. To further enhance computational efficiency, BitNet 1.58b extends the capabilities of standard BitNet by introducing ternary quantization (-1, 0, 1) instead of the conventional binary representation (-1, 1). This modification allows for selective weight deactivation, where weights assigned a value of zero can be ignored during computations, significantly reducing the number of operations required. As a result, matrix multiplications become more efficient, leading to faster inference speeds and lower power consumption which is a crucial factors for deploying large scale AI models in resource-constrained environments. Additionally, BitNet 1.58b employs absmean quantization for weight scaling, ensuring that weights are effectively mapped into the ternary space while maintaining model performance. Experimental results demonstrate the effectiveness of this approach, showing that a 13B BitNet 1.58b model not only achieves superior computational efficiency but also outperforms a 3B FP16 model in terms of latency, memory usage, and overall computational overhead. These advancements position BitNet 1.58b as a highly promising solution for large-scale AI deployment, particularly in energy-efficient and edge computing applications [6].

A.4 Dequantization

Dequantization is the process of restoring quantized values to higher precision formats (e.g., FP16 or FP32) when necessary. This is particularly useful for hybrid inference, where certain layers require full precision and for preserving numerical accuracy in sensitive computations.

This study will investigates the application of various quantization techniques to the remote sensing visual grounding task, assessing their impact on model accuracy degradation. The research aims to analyze the trade-offs between computational efficiency and performance loss, providing insights into the feasibility of deploying quantized models in resource-constrained environments.

References

[1] Y. Zhan, Z. Xiong, and Y. Yuan, "Rsvg: Exploring data and models for visual grounding on remote sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–13, 2023.

[2] L. Xiao, X. Yang, X. Lan, Y. Wang, and C. Xu, "Towards visual grounding: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2412.20206

[3] Y. Zhou, M. Lan, X. Li, Y. Ke, X. Jiang, L. Feng, and W. Zhang, "Geoground: A unified large vision-language model for remote sensing visual grounding," 2025. [Online]. Available: https://arxiv.org/abs/2411.11904

[4] K. Kuckreja, M. S. Danish, M. Naseer, A. Das, S. Khan, and F. S. Khan, "Geochat: Grounded large vision-language model for remote sensing," 2023. [Online]. Available: https://arxiv.org/abs/2311.15826

[5] Hugging Face, "Optimum: A Guide to Quantization," 2024, available at: texttthttps://huggingface.co/docs/optimum/en/concept_guides/quantization, Accessed: Feb. 10, 2025.

[6] M. Grootendorst, "A Visual Guide to Quantization," https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-quantization, 2024, accessed: Feb. 10, 2025.

[7] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "Llm.int8(): 8-bit matrix multiplication for transformers at scale," 2022. [Online]. Available: https://arxiv.org/abs/2208.07339

[8] C. Wang, Z. Wang, X. Xu, Y. Tang, J. Zhou, and J. Lu, "Q-vlm: Post-training quantization for large vision-language models," 10 2024.

[9] G.-S. Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, L. Zhang, and X. Lu, "Aid: A benchmark data set for performance evaluation of aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3965–3981, 2017.

[10] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. GIS '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 270–279. [Online]. Available: https://doi.org/10.1145/1869790.1869829

[11] S. Lobry, D. Marcos, J. Murray, and D. Tuia, "Rsvqa: Visual question answering for remote sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 12, pp. 8555–8566, 2020.

[12] L. Tunstall, L. von Werra, and T. Wolf, *Natural Language Processing with Transformers, Revised Edition*. O'Reilly Media, 2022. [Online]. Available: https://www.oreilly.com/library/view/natural-language-processing/9781098136789/