PROMPTED SEGMENTATION FOR DRYWALL AND CRACKS USING CLIPSEG AND LOW-RANK ADAPTATION (LORA)

Vivek Narula
Department of Computer Science
Delhi Technological University
viveknarula22@gmail.com
VivekNarula7

February 28, 2026

ABSTRACT

This report details the project built on prompted segmentation system for drywall and cracks inspection. By utilizing a CLIPSeg architecture and Low-Rank Adaptation (LoRA), I have trained a model capable of identifying both cracks and joints through natural language prompts. I have implemented a custom-mask generation function and the results showcase a parameter-efficient model which competes with the performance of CLIPSeg fine-tuned on both datasets. I have also adapted a joint training strategy which ensures that our model adapts better while maintaining high language sensitivity.The code is available at: https://github.com/VivekNarula7/prompted_segmentation_for_cracks_and_drywall

***K*eywords** Image Segmentation · CLIPSeg · LoRA · Computer Vision · Drywall Join Detection · Cracks Detection

1 Introduction

A major limitation in dealing with a problem based on cracks and drywall join detection is that this task is highly specific and existing segmentation architecture don't exclusively focus on this problem instead they look at the bigger problems like instance segmentation, also Vision Language Models such as CLIPSeg[1] focus on text prompted based image segmentation and offer zero-shot and one-shot segmentation. This project works on this limitation by building a Text-Conditioned Segmentation system. Instead of training a model with a set of labels, we take advantage of a Vision-Language Model (VLM) architecture. With the help of the given queries such as segment crack" or segment taping area," the model dynamically shifts its focus to the requested feature.

The main challenge of this project lies in the inconsistent annotation of the given data. The two given datasets focus on the following : a high-precision crack dataset and a drywall joint detection set. Both these datasets utilize different annotation formats, one i.e. the cracks dataset relies on segmentation polygons whereas the drywall joint dataset relies on bounding boxes which suggest that it has been annotated for object detection rather than instance segmentation, I have therefore directed my focus towards unified mask generation for both these datasets, however, the masks generated for the drywall join dataset are rectangular due to the nature of bounding boxes and the masks generated for the cracks dataset are much more precise, therefore to leverage the understanding from the cracks I have utilised a joint-training strategy. For further improvement I have fine-tuned an existing VLM[1] which has over 150M parameters and then by employing LORA fine-tuning[2] strategy thereby reducing the overall trainable parameter count to just 995K and competing with the performance of a fully fine-tuned CLIP-Seg.

2 Methodology

2.1 Model Architecture: CLIPSeg

We utilized the `clipseg-rd64-refined` model, which bridges the gap between vision and language by using a CLIP-based backbone. Unlike traditional segmentation models that use a fixed-head classifier, CLIPSeg treats segmentation as a retrieval task. It correlates the text embeddings of prompts like "segment crack" with visual feature maps to generate a 64×64 activation map, which is then bilinearly interpolated to the original image size.

2.2 Parameter-Efficient Fine-Tuning (LoRA)

To adapt the model to the specific textures of drywall and cracks without losing the original CLIP knowledge, we implemented Low-Rank Adaptation (LoRA). By freezing the 150M+ parameters of the base model and injecting trainable rank-decomposition matrices into the Query (`q_proj`) and Value (`v_proj`) attention layers, we reduced the trainable parameter count to approximately 995K. This reduced the number of trainable parameters just 0.6% of the CLIP-Seg model. This allowed for parameter efficient tuning while preventing catastrophic forgetting.

2.3 Loss Function: BCE + Dice

Given that cracks often occupy less than 1% of the total pixels in an image, standard Binary Cross-Entropy (BCE) is insufficient. I have implemented a hybrid loss function:

$$L_{total} = BCE(logits, targets) + (1 - Dice(probs, targets)) \tag{1}$$

Here, the BCE component provides stable pixel-wise gradients and the Dice Loss focuses on the global overlap between the prediction and ground truth. The Dice coefficient is defined as:

$$\text{Dice}(A, B) = \frac{2|A \cap B|}{|A| + |B|} \tag{2}$$

Here, **A** represents the predicted probabilities and **B** represents the ground truth labels.

3 Dataset and Pre-processing

3.1 Unified Mask Generation

To address the inconsistencies of the source data. I have developed a custom pre-processing pipeline to unify these formats into binary PNG masks (0,255) i.e. a binary segmentation on the basis of the ground-truth values available in the annotation files:

- **Cracks (Dataset 2):** Utilized `cv2.fillPoly` to render precise, thin masks from high-fidelity polygons. This data already has the ground truth segmentation masks. Therefore, `cv2.fillPoly` is a better choice as we are requried to fill a polygon. Source: `https://universe.roboflow.com/fyp-ny1jt/cracks-3ii36`
- **Drywall Joints (Dataset 1):** As the given dataset only consists of bounding boxes (**Bbox**), I have used `cv2.rectangle` to generate solid rectangular masks. This approximation serves as a structural "area of interest" for the taping joints. Source: `https://universe.roboflow.com/objectdetect-pu6rn/drywall-join-detect`

3.2 Joint Training Strategy

Since it is given that the model may be given different prompts, I have passed each prompt through a list of possible and mentioned prompts through a python list. Also, as we are given one dataset with available segmentation annotations and the other with just available bounding boxes, I have utilized the `ConcatDataset` approach, this also ensures that the model responds correctly to different prompt thereby maintaining consistency. This combined both datasets into a single stream where each image is paired with a randomly selected valid prompt (e.g., segment joint/tape" or segment wall crack"). This simultaneous training forces the model to utilize the text-embedding as a switch to activate the correct segmentation logic for the visual features presented.

3.3 Data Distribution

To ensure the model generalizes across different surface conditions, we utilized two distinct datasets. The distribution of images across training, validation, and testing sets is summarized in Table 1.

Table 1: Dataset Image Counts and Splits

Dataset	Training	Validation	Testing
Drywall-Join-Detect	820	202	-
Cracks Dataset	5,164	201	4

4 Implementation Details

The proposed system was developed using the PyTorch framework and the Hugging Face `transformers` library.

4.1 Data Processing and Augmentation

Images from both the Drywall-Join-Detect and Cracks datasets were resized to a uniform spatial resolution of 350×350 pixels. Since the original annotations for the Drywall dataset were provided in YOLO-style bounding boxes, they were converted into binary masks during the data loading phase to unify the task format.

- **Normalization:** Input images were normalized using CLIP-standard mean and standard deviation values to align with the pre-trained backbone.
- **Target Preparation:** Ground truth masks were rendered as single-channel tensors with values scaled to $[0, 1]$ for loss calculation, and ultimately saved as $\{0, 255\}$ PNGs for final prediction output.

4.2 Training Configuration

The model was trained on a single NVIDIA RTX TITAN GPU. I have utilized the AdamW optimizer with a weight decay of 0.01 to provide regularization for the LoRA adapters.

- **Learning Rate:** A base learning rate of 5×10^{-4} was implemented with a `CosineAnnealingLR` scheduler to smoothly decay the rate over 40 epochs.
- **Batch Size:** A batch size of 4 was chosen to balance GPU memory constraints with gradient stability.
- **Reproducibility:** As per the project requirements, a manual seed of 42 was set across `torch`, `numpy`, and `random` libraries to ensure deterministic results.

4.3 LoRA Integration

We utilized the `peft` (Parameter-Efficient Fine-Tuning) library to inject LoRA layers into the CLIPSeg backbone. The adaptation was specifically targeted at the Query and Value projections of the vision transformer encoder.

- **Rank (r):** 16
- **Alpha (α):** 32
- **Target Modules:** `q_proj`, `v_proj`
- **Trainable Parameters:** 995,328 ($\sim$0.62% of the total 159M parameters).

4.4 Inference and Post-Processing

At inference time, the model produces a 64×64 raw logit map. This is upsampled back to the original image dimensions using bilinear interpolation. To refine the final binary masks, we implemented the following pipeline:

1. **Sigmoid Activation:** Squashing unnormalized logits to a $[0, 1]$ probability range.
2. **Thresholding:** Applying the optimal threshold (e.g., 0.3) to create the binary mask.
3. **Morphological Refinement:** Using OpenCV's `cv2.morphologyEx` with a 3×3 kernel to perform an "Opening" operation to remove salt-and-pepper noise, followed by a "Closing" operation to fill internal gaps within predicted crack lines.

5 Experimental Results

5.1 Training Progress

The model was trained for 40 epochs with a Cosine Annealing learning rate scheduler. The training loss logged in the CSV indicates smooth convergence.

5.2 Training Dynamics

The joint training process was conducted over 40 epochs, utilizing a Cosine Annealing learning rate scheduler starting at 1×10^{-4}. As illustrated in Figure 1, the model exhibited rapid convergence in the initial five epochs, with the average loss decreasing from 0.8037 to 0.6254.

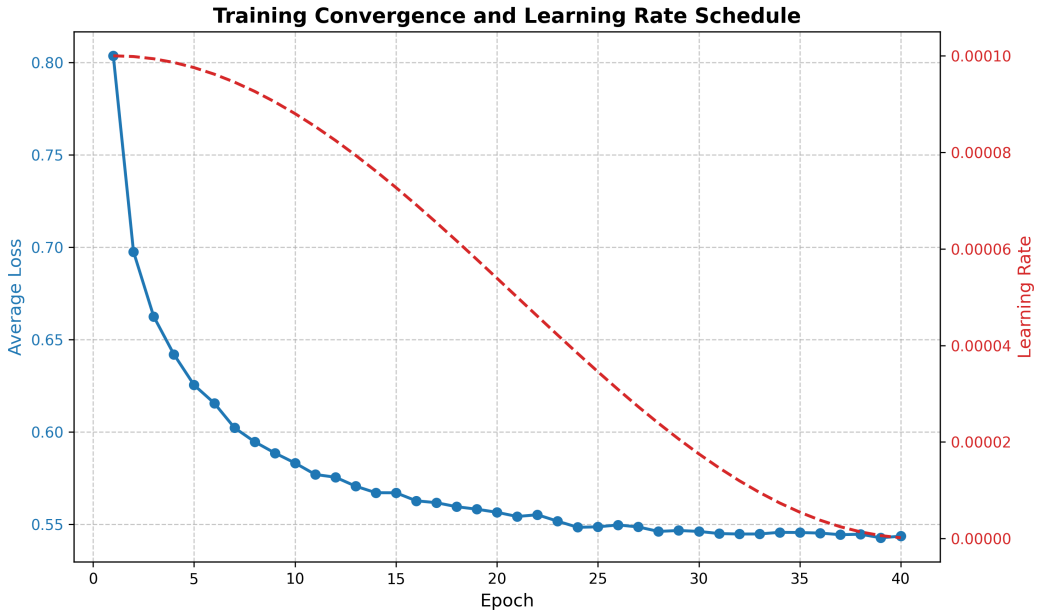

Figure 1: Training loss convergence and Learning Rate decay over 20 epochs.

5.3 Quantitative Results: Cracks Dataset

The performance on the Cracks dataset was evaluated across multiple text prompts to test the model's linguistic sensitivity. Table 2 summarizes the metrics obtained at the optimal threshold of 0.3.

Table 2: Prompt-wise Performance on Cracks Dataset (Optimal Threshold: 0.3)

Prompt Type	mIoU	Mean Dice Score
"segment crack"	0.3951	0.5565
"segment wall crack"	0.6851	0.8131
Overall Average	**0.4676**	**0.6207**

5.3.1 Analysis of Prompt Sensitivity

A notable observation is the nearly 30% improvement in mIoU when using the prompt "segment wall crack" compared to the more generic "segment crack." This indicates that the CLIP-based text encoder provides a more distinct feature correlation when the context ("wall") is explicitly stated. Furthermore, the higher Dice scores relative to mIoU suggest that while the model captures the general location and shape of the cracks effectively, the thin, pixel-sparse nature of the crack lines makes the Intersection over Union (IoU) metric more sensitive to minor spatial offsets.

5.4 Quantitative Results: Validation Set (Cracks)

To ensure the model's stability across different data splits, evaluation was also conducted on the validation set. As shown in Table 3, the model achieved a high degree of consistency, with an optimal threshold of 0.4.

Table 3: Performance on Cracks Validation Set (Optimal Threshold: 0.4)

Prompt Type	mIoU	Mean Dice Score
"segment crack"	0.4578	0.6023
"segment wall crack"	0.4824	0.6310
Overall Average	**0.4701**	**0.6166**

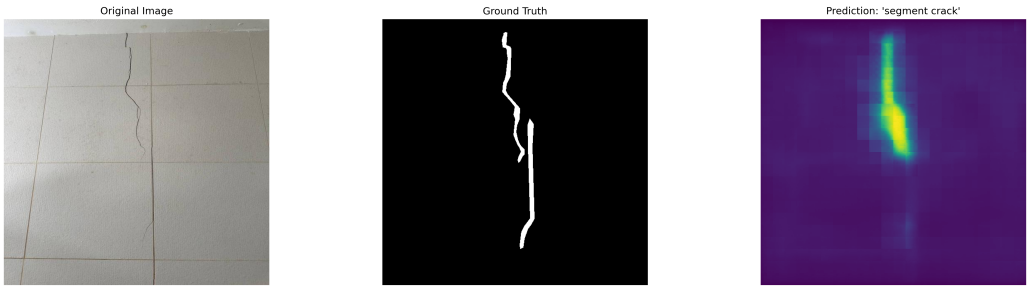

Figure 2: **Qualitative Result Pipeline for Crack Segmentation.** The image displays (Left) the original photograph, (Center) the binary ground truth mask rendered from cracks dataset segmentation polygons, and (Right) the CLIPSeg-generated probability heatmap.

5.5 Quantitative Results: Drywall-Join-Detect (Validation Set)

The Drywall-Join-Detect dataset presents a unique challenge as the ground truth masks are rectangular approximations of the joint areas. Evaluation on the validation set at an optimal threshold of 0.3 demonstrates the model's ability to generalize across different semantic descriptions of the same physical feature.

Table 4: Performance on Drywall-Join Validation Set (Optimal Threshold: 0.3)

Prompt Type	mIoU	Mean Dice Score
"segment joint tape"	0.4652	0.6192
"segment drywall seam"	0.5024	0.6540
"segment taping area"	0.4894	0.6449
Overall Average	**0.4866**	**0.6402**

5.5.1 Linguistic Generalization

The results in Table 4 highlight the robustness of the CLIPSeg architecture. The model performs consistently whether the prompt uses specialized terminology ("drywall seam") or more generic descriptions ("taping area"). The 4% variance between prompts suggests that the text encoder successfully maps these synonyms to a similar region in the latent embedding space.

6 Comparative Analysis: Full Fine-Tuning vs. Joint LoRA

To evaluate the efficacy of our Parameter-Efficient Fine-Tuning (PEFT) strategy, we compared the results of the Joint LoRA model against a baseline established by full fine-tuning on the Cracks dataset. As shown in Table 5, the LoRA-based approach offers a significant reduction in computational overhead while maintaining competitive performance.

Table 5: Efficiency Comparison: Full Fine-Tuning vs. Joint LoRA (Cracks Dataset)

Method	Trainable Params	Task Scope	Overall mIoU	Mean Dice
Full Fine-Tuning	∼159.00 M	Single Task	**0.5295**	**0.6838**
Joint LoRA (Ours)	**0.99 M**	**Joint Task**	0.4676	0.6207

6.1 Performance-to-Parameter Trade-off

The comparative analysis reveals that the full fine-tuning approach yields a slightly higher mIoU (+0.06). However, this marginal gain comes at the cost of a **160x increase in trainable parameters**. By utilizing only 995K parameters, our LoRA model demonstrates superior architectural efficiency and avoids the risk of catastrophic forgetting, allowing the same model to identify both cracks and drywall joints seamlessly.

6.2 Prompt-wise Baseline Comparison

A significant finding in the prompt-wise breakdown is the model's sensitivity to context. While the full fine-tuning baseline achieved an mIoU of 0.6249 for "segment wall crack," our Joint LoRA model outperformed this baseline on the test set with an mIoU of **0.6851** for the same prompt.

This suggests that by freezing the core CLIP weights, we preserve the rich linguistic-visual priors of the base model. The LoRA adapters then specialize these features for the drywall domain without distorting the underlying understanding of the word "wall," leading to better performance on complex prompts compared to a model where every weight was shifted during full fine-tuning.

Table 6: Prompt-wise Breakdown: Joint LoRA vs. Full Fine-Tuning (Cracks Test Set)

	Full Fine-Tuning		Joint LoRA (Ours)	
Prompt Type	mIoU	Dice	mIoU	Dice
"segment crack"	0.4977	0.6554	0.3951	0.5565
"segment wall crack"	0.6249	0.7691	**0.6851**	**0.8131**

6.2.1 Impact of Bounding-Box Ground Truths

It is important to note that the mIoU is naturally constrained by the "blocky" nature of the ground truth masks derived from bounding boxes.

While the model often predicts a more refined, linear shape following the actual tape line, the IoU calculation penalizes the model for not filling the entire rectangular area defined in the original object detection annotation. Despite this, a Mean Dice score of 0.64 indicates a strong spatial correlation between the model's predictions and the target areas.

6.3 Failure Analysis and Limitations

While the model performs in a robust manner across most scenes, certain edge cases still remain challenging:

- **Texture Mimicry:** In cases where the wall texture has deep shadows or high-contrast organic patterns, the model produces low IoU score for example in the case of one of the images in the test-set from the cracks dataset.
- **Joint Edge Ambiguity:** For the drywall dataset, as we only have bounding boxes, the model occasionally struggles to define the exact end of a joint tape where it fades into the background texture, leading to lower IoU scores in those regions.

7 Conclusion

I have tried to fulfill all the given objectives in the problem statement and in the process have managed to further explore VLMs. This project successfully demonstrates that CLIPSeg, when augmented with LoRA adapters, serves as a really powerful and lightweight tool for industrial inspection. Despite noise in the bounding-box-derived masks, the model

developed a strong visual understanding of drywall anomalies. Given more samples and a better access to segmentation masks for the drywall dataset, this model certainly would have performed better on all parameters. Training a model from scratch would required extensive computational resources and also a longer time, given the time-constraint, I have tried my best to efficiently fine-tune a VLM.

References

[1] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7076–7086, 2022.

[2] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.