https://doi.org/10.31436/iiumej.v27i2.4124

PromptLessSAM: From Foundational Model to Domain Expert via Lightweight Decoder Adaptation for Crack Segmentation

MOHAMMED AL-MUSTAFA*, ISRAA H. ALI

Software Department, College of Information Technology, University of Babylon, Babylon, Iraq

**Corresponding author: mohammedalmustafa.hendo@student.uobabylon.edu.iq*

(Received: 10 January 2026; Accepted: 20 April 2026; Published online: 10 May 2026)

ABSTRACT: Maintaining infrastructure such as roads and bridges is very important for safety. This is called Structural Health Monitoring (SHM). A key part of SHM is automatically finding cracks in pavements. Recent foundation models, such as the Segment Anything Model (SAM), can segment a wide range of object types. However, they are not well-suited for specific tasks such as detecting thin cracks in complex environments. They also require user input to function, making them unsuitable for fully automatic inspection systems. Therefore, this paper proposes PromptLessSAM to transform the general SAM model into a domain expert for crack segmentation. We use a lightweight method in which we freeze the powerful image encoder of SAM, except for the encoder's neck, which remains trainable, and we add a new, small decoder. Our model is very efficient and has only about 1.1 million trainable parameters. We trained and tested our model on the public Crack500 dataset. The experimental results show that our model achieved high performance (a Dice score of 72.7% and an IoU of 57.11% on Crack500), outperforming many state-of-the-art models, including TransUNet and PaveSAM. Also, our model showed good generalization capabilities on the DeepCrack dataset (Dice score of 74.7% and IoU of 62.43%), which it did not see during training. PromptLessSAM provides a new, efficient, and effective method. It shows how to adapt large foundation models for a specific task. Our work provides a strong, high-performance solution trained on a very small number of parameters.

ABSTRAK: Penyelenggaraan infrastruktur seperti jalan raya dan jambatan sangat penting bagi tujuan keselamatan. Ini dikenali sebagai Pemantauan Kesihatan Struktur (Structural Health Monitoring, SHM). Salah satu komponen utama pada SHM ialah mengesan rekahan pada turapan secara automatik. Model asas terkini, seperti Model Sesuatu Bahagian (SAM), boleh melakukan segmentasi pelbagai objek. Namun, model ini kurang berkesan bagi tugas khusus seperti mengesan rekahan yang nipis pada persekitaran kompleks. Selain itu, SAM memerlukan arahan daripada pengguna untuk berfungsi, menjadikannya kurang sesuai bagi sistem pemeriksaan sepenuhnya automatik. Oleh itu, kajian ini mencadangkan PromptLessSAM untuk menukarkan model SAM yang umum kepada model pakar domain bagi segmentasi rekahan. Kajian ini menggunakan kaedah yang ringan dengan membekukan pengekod imej SAM yang berkuasa, kecuali pada bahagian 'leher' pengekod yang kekal dilatih, dan menambah sebuah penyahkod baharu yang kecil. Model ini sangat cekap dan hanya mempunyai kira-kira 1.1 juta parameter boleh latih. Model ini dilatih dan diuji menggunakan set data awam Crack500. Keputusan eksperimen menunjukkan model ini mencapai skor prestasi tinggi (iaitu skor Dice sebanyak 72.7% dan IoU sebanyak 57.11% pada set data Crack500), mengatasi banyak model tercanggih (state-of-the-art) seperti TransUNet dan PaveSAM. Selain itu, model ini menunjukkan keupayaan generalisasi yang baik pada set data DeepCrack (skor Dice 74.7% dan IoU 62.43%), yang tidak pernah dilihat semasa latihan. PromptLessSAM menyediakan kaedah baharu yang cekap dan berkesan, serta menunjukkan cara menyesuaian model asas berskala besar bagi tugas khusus. Kajian ini

https://doi.org/10.31436/iiumej.v27i2.4124

menawarkan penyelesaian kukuh dengan prestasi tinggi namun dilatih menggunakan bilangan parameter paling rendah.

KEYWORDS: *Crack Semantic Segmentation, Segment Anything Model (SAM), Deep Learning, Structural Health Monitoring.*

1. INTRODUCTION

The regular inspection and maintenance of infrastructure, such as roads, bridges, and buildings, are essential for public safety and economic stability. This process is often called Structural Health Monitoring (SHM) [1]. Pavement distress, especially cracking, is a common and early sign of structural damage. Detecting these cracks early allows for timely repair, which can prevent bigger problems and save a lot of money [2]. For many years, this inspection was done manually by trained inspectors. This manual method is slow, costs a lot, can be dangerous for the inspectors, and the results can differ depending on the person [3].

To solve these problems, researchers have developed automatic methods using computer vision and deep learning [2]. Models like U-Net [4] and its variations have shown good results for semantic segmentation. This means assigning a label to every pixel in an image. More advanced models have also been used [5]. However, these deep learning models have two main challenges. First, they often need a very large number of parameters, making them heavy and slow. For example, some models use over 100 million parameters [6]. This makes them hard to use on small devices or for real-time inspection. Second, they need a large, high-quality dataset with pixel-level labels, which is very difficult and expensive to create [7].

Recently, a new type of model, called a "foundation model," has emerged. The Segment Anything Model (SAM) from Meta AI is a powerful example [8]. SAM was trained on a huge dataset of 11 million images and can segment almost any object in any image. The main problem is that SAM is a generalist model. It was not trained for specific tasks, such as pavement crack detection. As shown in other studies, SAM does not perform well on pavement cracks, which are very thin and have complex patterns, unlike the common objects in its training data [9]. Furthermore, SAM needs a "prompt" from the user, such as a point or a box, to determine what to segment. This makes it unsuitable for a fully automatic crack detection system.

Lightweight crack segmentation is important for real-world deployment because many inspection scenarios rely on edge platforms such as mobile robots, UAVs, or handheld devices, where memory, power, and latency are limited. Recent studies explicitly target this constraint; for example, CrackESS [10] validates crack segmentation in an edge setting and demonstrates deployment on a climbing robot platform using embedded hardware acceleration. Although foundation models such as SAM provide strong general representations, their large parameter size and compute cost make direct fine-tuning and deployment difficult in these constrained environments. Therefore, parameter-efficient adaptation is necessary because it can transfer foundation-model knowledge to the crack domain while updating only a small subset of parameters, thereby reducing training cost and making practical deployment more realistic.

This paper presents a method to solve these problems, called PromptLessSAM. We change the SAM foundation model from a generalist into a domain expert for crack segmentation. We do this by adapting it in a lightweight way. Our main idea is to keep most of the powerful, pre-trained image encoders of SAM frozen. This encoder already knows how to see rich features in an image. We then remove the original prompt encoder and mask decoder. We add our own new, very small decoder. These lightweight decoder and encoder neck layers are the only parts

https://doi.org/10.31436/iiumej.v27i2.4124

of the model that we train. This method is a form of Parameter-Efficient Fine-Tuning (PEFT) [11]. This has two benefits. First, it makes the model "prompt-less" because the decoder is trained to only find cracks. Second, it is efficient training. We only need to train about 1.1 million parameters. We show that our PromptLessSAM model outperforms many larger, more complex models on the Crack500 dataset. Our work provides an efficient way to adapt large foundation models for specific, important tasks.

The main contributions of this work are as follows:

- PromptLessSAM is proposed as a fully automated (prompt-less) crack segmentation approach by adapting the SAM image encoder with a lightweight crack-specific decoder.
- A parameter-efficient training strategy is applied by freezing the main encoder blocks and training only the encoder neck and decoder, which keeps the number of trainable parameters small.
- Competitive results are achieved on Crack500, and generalization to an unseen dataset (DeepCrack) is demonstrated without requiring any prompt input during inference.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 details our proposed methodology. Section 4 presents the experiments and results. Section 5 provides a comprehensive discussion of the findings and their significance. Finally, Section 6 concludes the paper and outlines directions for future research.

2. RELATED WORKS

The field of automated crack segmentation has seen much research, moving from traditional image processing to advanced deep learning [2]. Early methods used techniques like edge detection, thresholding, and morphological operations [12]. These methods are simple but are not robust. They fail in poor lighting, with shadows, or with different pavement textures. These traditional methods are often not reliable for complex road conditions [12]. In this review, representative works published are considered, covering CNN-based, Transformer-based, SAM-based, state-space (Mamba), and weakly/unsupervised approaches, and Table 1 summarizes their datasets, advantages, and limitations.

To overcome these issues, researchers turned to deep learning and convolutional neural networks (CNNs) [12]. The U-Net [4] architecture became very popular for this task. It uses an encoder-decoder structure with skip connections, which helps to get precise segmentation. Many other models are based on the U-Net. Other works, like DeepCrack [7], use a hierarchical feature learning approach. While these CNN-based methods perform well, they are often trained from scratch and require a large amount of labeled data to learn crack features.

More recently, Vision Transformers (ViT) [13] have shown great success in computer vision. Transformers use self-attention to capture long-range dependencies, which can be useful for segmenting long, thin cracks [14]. Models like TransUNet [6] combine the benefits of U-Net and Transformers. They use a Transformer as the encoder to learn global features and a CNN-based decoder to produce high-resolution segmentation [14]. However, these Transformer-based models are often very large, with many trainable parameters, like TransUNet, which has 101M parameters [6]. This makes them computationally expensive to train and deploy.

The release of the SAM [8] created a new direction for segmentation. SAM is a foundation model for segmentation. It can segment anything based on prompts. However, its zero-shot performance on specialized tasks like crack segmentation is poor [9]. Some recent research has

https://doi.org/10.31436/iiumej.v27i2.4124

tried to adapt SAM for specific domains. For example, PaveSAM [9] tried to use SAM for pavement distress but found its baseline performance was very low. CrackESS [10] proposed a system using YOLOv8 for self-prompting and a fine-tuned SAM. Other methods aim to make SAM smaller, such as FasterSAM [15]. Our work is different because we create a truly prompt-less model by replacing the decoder with a new, lightweight module, making it efficient and fully automatic.

Table 1. Related works summary (crack segmentation and SAM adaptation).

Model / Approach	Dataset Used	Key Finding	Advantage	Limitations / Evaluation metrics and accuracy
U-Net (CNN encoder–decoder) [4,18]	Crack500	Strong baseline with skip connections	Simple and stable	Full supervision; Eval: Precision (P), Recall (R), F1/Dice, IoU
DeepCrack (hierarchical CNN) [7]	Crack500, DeepCrack	Multi-level features improve crack detail	Good thin-crack sensitivity	Needs pixel labels; Eval: P, R, F1, IoU
TransUNet (Transformer + CNN) [6,19]	Crack500	Global context + decoder refinement	Better long-crack continuity	High compute; Eval: F1/Dice, IoU
SwinCrack (Swin Transformer) [18,20]	Crack500	Self-attention helps long-range crack modeling	High precision behavior	Heavier than CNN; Eval: P, R, F1, mIoU
VM-UNet32 (Mamba/SSM) [18,21]	Crack500	Mamba encoder improves context capture	Competitive performance	Complex design; Eval: P, R, F1, mIoU
DTrC-Net (hybrid) [19]	Crack500	Hybrid design for crack segmentation	Balanced accuracy/complexity	Higher params; Eval: mDS (Dice), mIoU
UP-CrackNet (unsupervised) [16]	Crack500, DeepCrack	Unsupervised via adversarial restoration	No pixel labels required	Sensitive to textures; Eval: Accuracy, P, R, F1, IoU
WP-CrackNet (weakly supervised) [17]	Crack500, DeepCrack	Weak supervision via adversarial learning	Reduced annotation cost	CAM incompleteness risk; Eval: Accuracy, P, R, F1, IoU
PaveSAM (SAM + box prompts) [9]	Crack500, Custom pavement distress dataset	SAM adaptation using box prompts for distress segmentation	Improves over zero-shot SAM	Requires prompts; Eval: P, R, F1, IoU
CrackESS (YOLO self-prompt + SAM) [10]	Crack500	Self-prompting (YOLO) + SAM-based segmentation for edge systems	Higher automation	Still prompt-based; Eval: P, R, F1, IoU, FPS

In addition to fully supervised CNN and Transformer models, recent studies investigate methods to reduce dependence on expensive pixel-level labels. UP-CrackNet [16] introduces an unsupervised pixel-wise framework that uses adversarial image restoration to learn crack features without direct crack supervision. WP-CrackNet [17] extends this line of work by using a collaborative adversarial learning framework for end-to-end weakly supervised road crack detection, showing that weaker annotations can still yield usable segmentation outputs. Furthermore, lightweight designs based on state-space models (SSMs) and Mamba have been explored to capture long-range crack patterns at reduced computational cost compared with heavy Transformers [18,19]. Table 1 summarizes representative recent studies, including

https://doi.org/10.31436/iiumej.v27i2.4124

datasets, key findings, advantages, and limitations, and evaluates them using the evaluation metrics to clarify the research gap addressed by our prompt-less SAM adaptation.

3. METHODOLOGY

In this section, we first revisit SAM and then introduce the task addressed in this paper: SAM for Pavement Distress Segmentation. Next, we delve into how to achieve the pipeline of the proposed framework. Finally, we outline the training and evaluation processes employed in our approach.

3.1. Preliminary

3.1.1. Revisit of Segment Anything Model

The SAM [9] is a foundation model designed for promptable segmentation. It was trained on the large SA-1B dataset. This training allows SAM to have strong zero-shot generalization capabilities. As shown in Figure 1, SAM's architecture has three main parts. First is a powerful image encoder, which is a ViT [8]. This encoder processes the input image and creates a rich feature embedding. Second is a prompt encoder. This part takes user inputs, such as points, boxes, or text, and converts them into an embedding. Third is a lightweight mask decoder. This decoder takes the image embedding and the prompt embedding and produces the final segmentation mask. This design is flexible, but it always requires a prompt to tell the model what to segment.

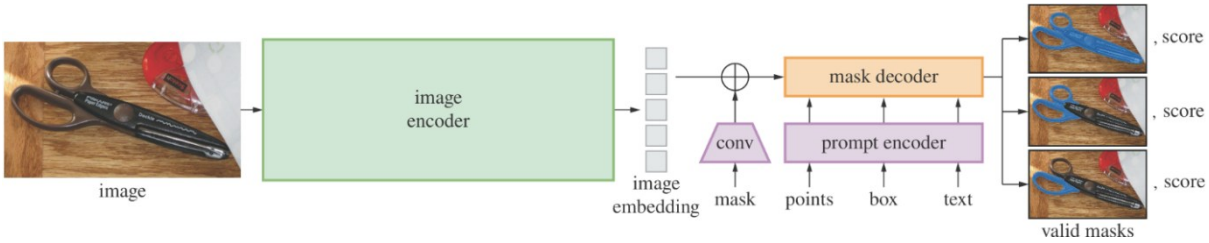


Figure 1. The architecture of the original SAM, showing the Image Encoder, Prompt Encoder, and Mask Decoder.

3.1.2. SAM on Pavement Distress Segmentation

While SAM is effective for general segmentation, it struggles with specialized tasks such as pavement crack segmentation. This is because pavement distresses are very different from the "natural" objects in SAM's training data [8]. Cracks are often very thin, have low contrast with the road surface, and form complex, irregular networks [7]. When SAM is applied to crack images without any task-specific training, its performance is very poor.

Research by Owor et al. on PaveSAM [9] evaluated SAM's zero-shot performance on crack datasets. Their results showed that SAM alone is not suitable for this task. For example, on the Crack500 dataset, their experiments showed that SAM achieved a Dice score of only 0.137. This is much lower than even older models such as U-Net [4]. This poor performance shows that SAM, in its original form, cannot be used as a reliable tool for automatic pavement inspection. It does not understand the specific features of cracks. Therefore, adaptation is necessary to make it useful for this domain. SAM does not produce good results on pavement distress, as shown in Figure 2.

https://doi.org/10.31436/iiumej.v27i2.4124

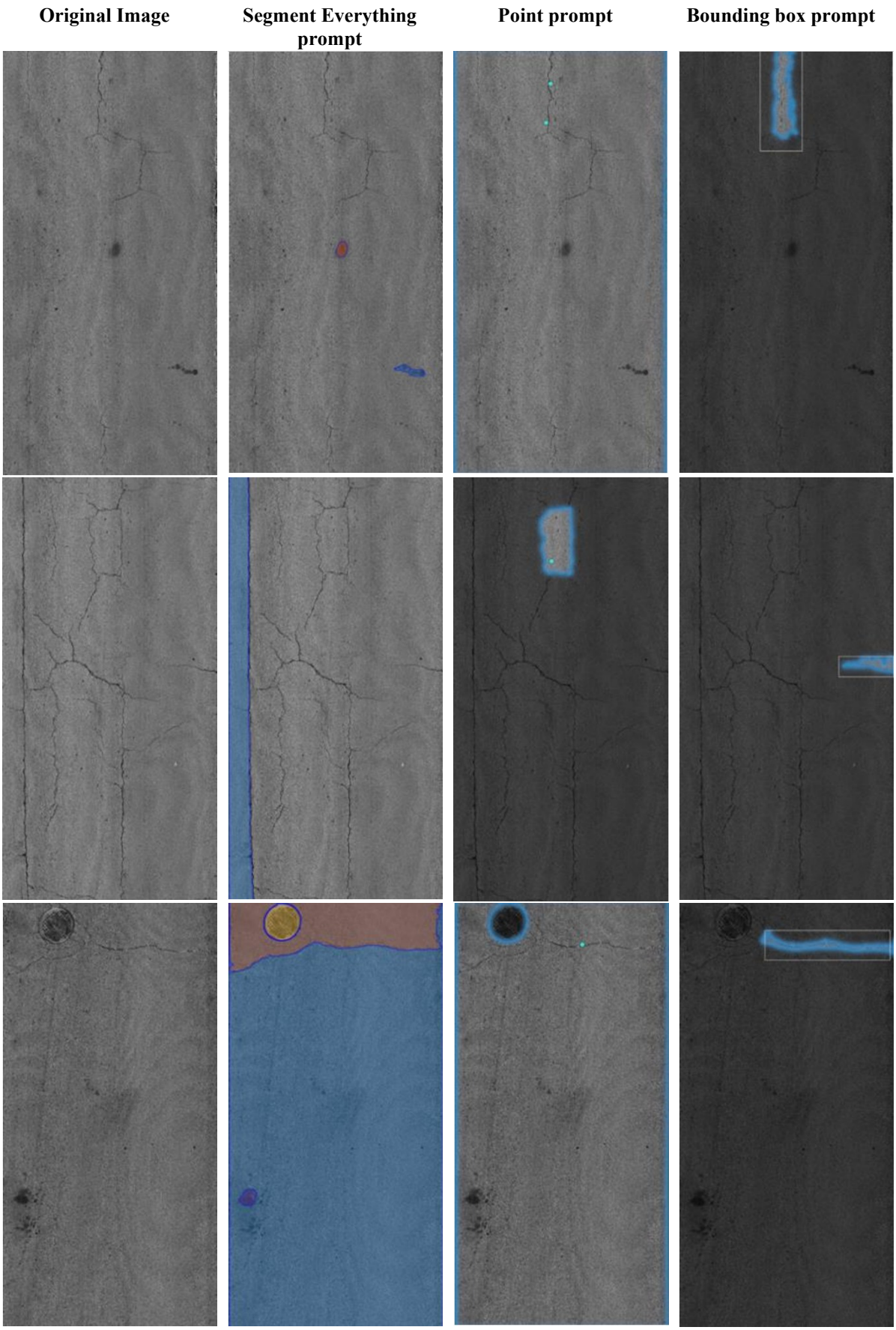


Figure 2. Examples of SAM's poor zero-shot performance on pavement crack images [9].

https://doi.org/10.31436/iiumej.v27i2.4124

3.2. Proposed Method (PromptLessSAM)

To address poor performance and the need for prompts, we propose PromptLessSAM. Our goal is to transform SAM from a general-purpose, promptable model into an automatic, specialized "expert" for crack segmentation. We do this by changing its architecture. The key idea is to leverage the powerful pre-trained SAM image encoder while replacing the other parts. The architecture of our proposed model is shown in Figure 3.

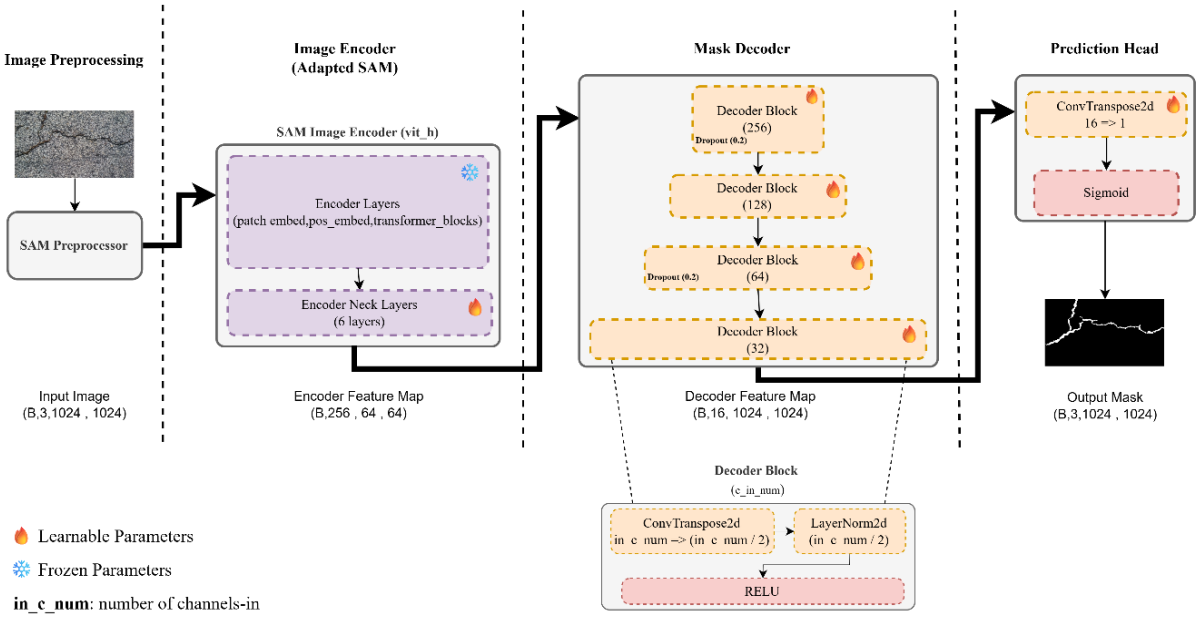


Figure 3. The proposed architecture of PromptLessSAM.

3.2.1. Model Framework Overview

Our framework has two main components: a partially frozen image encoder and a lightweight, trainable decoder. We take the pre-trained image encoder from SAM. This encoder comprises a main ViT-H backbone and a "Neck" module. We freeze the ViT-H backbone weights. This is important because this part has billions of parameters and has already learned powerful, general-purpose features. By freezing it, we save a huge amount of computation and prevent overfitting.

We then allow the "Neck" of the SAM encoder to be trainable. This part of the encoder helps to process the features from the ViT backbone. By fine-tuning it, we adapt these general features to the specific task of crack detection. We completely remove SAM's original prompt encoder and mask decoder. In their place, we attach a new, lightweight decoder. This decoder is also trainable and learns to interpret the adapted features from the Neck to find *only* cracks. Because the Neck and decoder are trained only on crack images, the model automatically outputs a crack mask for any input image. This makes our model fully automatic.

3.2.2. Image Preprocessing

To use the powerful SAM image encoder, it is necessary to prepare the input images correctly. The encoder is a ViT and was pre-trained on images of a specific size and format. If we give the encoder different images, it will not produce good features. Therefore, our model uses the standard preprocessing function from the original SAM [8]. Preprocessing resizes every input image from the dataset to a fixed resolution of 1024x1024 pixels. After resizing, the function also normalizes the pixel values using the mean and standard deviation used by

https://doi.org/10.31436/iiumej.v27i2.4124

the SAM model during training. This step is very important to make sure all images are in the correct format for the frozen ViT backbone.

3.2.3. PromptLessSAM Architecture

Our proposed architecture comprises a modified SAM encoder and a new lightweight decoder, described in detail in this section.

a. Image Encoder

The image encoder is the backbone of our model. We use the ViT-H encoder from the original SAM. This encoder, as shown in Figure 3, can be divided into two parts: the main Image Encoder (ViT-H) backbone and a Neck. The ViT-H backbone processes the input image using a series of Transformer blocks to create a rich feature embedding [13]. In our method, this large backbone is completely frozen.. Its weights are not updated during training.

The output of the frozen ViT-H backbone is then passed into the Neck. The Neck job refines features from the backbone. Unlike the backbone, we make this Neck trainable. This allows the model to learn to adapt general features to the specific, fine-grained details of cracks. The output of this trainable Neck is then passed to our new decoder.

b. Decoder Block

The Decoder Block is the fundamental building block of our lightweight decoder. Its main purpose is to take the encoder's feature maps and progressively upsample them to reconstruct the full-resolution mask. As shown in Figure 3, each Decoder Block is a sequence of layers. The block starts with a Transposed Convolution (ConvTranspose2d) layer. This layer performs upsampling, which doubles the spatial dimensions of the feature map (e.g., from 64×64 to 128×128) while reducing the number of feature channels. The transposed convolutions use a 2×2 kernel with a stride of 2 and no padding.

After the upsampling, we apply a LayerNorm2d layer. This is the same 2D layer normalization used in the original SAM architecture, which normalizes the features to stabilize training. This is followed by a Rectified Linear Unit (ReLU) activation function, which introduces non-linearity to help the model learn complex crack patterns. A dropout layer is also added to some of the blocks. This is a regularization technique that helps prevent overfitting during training.

c. Lightweight Decoder

The core of our contribution is the lightweight decoder, which is built by stacking the Decoder Blocks described in the previous section. This decoder is designed to be small and efficient. It has a U-Net-like structure that takes the low-resolution feature map from the encoder's trainable Neck and gradually upsamples it. The SAM encoder output is 256×64×64. Our decoder uses four Decoder Blocks in sequence.

These blocks double the resolution and halve the channels at each step (256×64×64 → 128×128×128 → ... → 16×512×512).

d. Prediction Head

After the last Decoder Block, the feature map has a resolution of 1024x1024 with 16 feature channels. The final part of our network is the Prediction Head. This part converts these features into the final segmentation mask. It consists of a single 2D transposed convolution layer, which acts as a 1x1 convolution. This layer has a kernel size of 1 and a stride of 1.

https://doi.org/10.31436/iiumej.v27i2.4124

The purpose of this layer is to reduce the number of feature channels from 16 to 1. This operation creates a single-channel feature map. This map is then passed through a Sigmoid activation function. The Sigmoid function is important because it maps all values to the [0,1] interval. The result is the final probability map, in which each pixel value represents the model's confidence that the pixel belongs to a crack.

3.3. Dataset

We used the public Crack500 dataset [9] for training and testing our model. This dataset is a standard benchmark for pavement crack segmentation, as used in many studies [9], [10]. It contains 500 images of size around 2000 1500 pixels of pavement cracks. The images are high-resolution and cover a range of crack types and pavement conditions. The dataset is divided into 250 training images, 50 validation images, and 200 test images. It includes high-quality pixel-level ground-truth masks. We also use the DeepCrack dataset [7] to test our model's generalization ability. The dataset consists of 537 pixel-level annotated images of cracks on concrete and asphalt surfaces across various scenes and scales, with a resolution of 544 × 384 pixels [17]. In a common split, 300 images are used for training, and 237 images are used for testing [16]. In our experiments, we do not train or tune the model on DeepCrack and use it only as an unseen test set to evaluate cross-dataset generalization.

3.4. Training Strategy

3.4.1. Implementation

Our model was implemented using the PyTorch [22] library and trained on a cloud platform (RunPod) using NVIDIA GPU (RTX 5090). The model uses the Adam [23] optimizer with beta values of (0.9, 0.999).

Our training process was done in steps. We first trained the model using binary cross-entropy (BCE) with a learning rate of 0.01. We found that the best model checkpoint on the testing dataset was achieved at epoch 12. We then reloaded this best checkpoint (epoch 12) and continued training for another 25 epochs. For this second phase of training, we used Dice-BCE loss and the same learning rate (0.01) to further fine-tune the model's performance.

3.4.2. Loss Function

Choosing the right loss function is very important for segmentation tasks, especially with imbalanced data. Crack segmentation is highly imbalanced because most of the image is "not crack" (background) and only a few pixels are "crack".

Binary Cross-Entropy (BCE) [24] Loss is a popular loss function; it performs well for pixel-level classification. BCE loss is used to minimize pixel-level errors by penalizing the difference between predicted probabilities and the actual class labels, encouraging accurate predictions. BCE loss is defined as:

$$Loss_{BCE}(y, \hat{y}) = -(y\, log(\hat{y}) + (1 - y)\, log(1 - \hat{y})) \quad (1)$$

where $\hat{y}$ is the predicted value by the prediction model, and y is the actual value.

Dice Loss [25] is very good for imbalanced data because it measures the overlap between the predicted mask and the true mask. The Dice coefficient measures this overlap. Dice loss is defined as follows:

$$Loss_{Dice} = 1 - \frac{2\Sigma_{i=1}^{n} p_i y_i}{\Sigma_{i=1}^{n} p_i^2 + \Sigma_{i=1}^{n} y_i^2} \quad (2)$$

where y_i is the actual pixel value, and p_i is the predicted pixel value.

https://doi.org/10.31436/iiumej.v27i2.4124

Our final loss function is the sum of these two losses:

$$Loss_{final} = Loss_{BCE} + Loss_{Dice} \quad (3)$$

This combined loss helps the model produce accurate pixel classifications (from BCE) while also ensuring high overlap with the true cracks (from Dice Loss).

3.4.3. Evaluation Methods

To evaluate the performance of the PromptLessSAM model, we use several standard semantic segmentation metrics. These metrics are calculated using the True Positives (TP), False Positives (FP), and False Negatives (FN) from the test set.

- Precision [26]: This measures how many of the pixels we predicted as "crack" were actually cracks.

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

- Recall [27]: This measures how many of the "crack" pixels in the image we were able to find.

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

- Dice Score (F1-Score) [7]: This is the harmonic mean of Precision and Recall. It is a very common metric for segmentation.

$$F1\ score = \frac{2*Precision*Recall}{Precision+Recall} \quad (6)$$

- Intersection over Union (IoU) [26]: This is another very common metric that measures the overlap between the predicted and true masks.

$$IoU = \frac{TP}{TP+FP+FN} \quad (7)$$

We use these four metrics to compare our model to other methods.

4. EXPERIMENTS AND RESULTS

4.1. Implementation Details

As mentioned in Section 3.4.1, the model was built in PyTorch [22] and trained using an Adam optimizer [23]. The Learning Rate (LR) was set to 0.01 because this value is commonly used as an initial LR in crack segmentation training, especially when the training schedule is short or when only part of the model is optimized. For example, the importance-enhanced Mamba-based crack segmentation network reports using the Adam optimizer with an initial learning rate of 0.01 during training. [18] In addition, UP-CrackNet [16] reports an initial learning rate of 0.01 for supervised crack segmentation settings (with poly adjustment), indicating that this LR scale is practical for crack segmentation optimization. In our case, only the encoder neck and lightweight decoder are trainable, so a moderate LR helps the model converge efficiently. A much smaller LR reduces the update magnitude and can slow convergence.

The training was performed on a RunPod [28] cloud computing platform, which provided the necessary GPU resources. The model uses a frozen SAM ViT image encoder and a custom lightweight decoder. The total number of trainable parameters in our model is only 1,093,345 (~1.1 million).

https://doi.org/10.31436/iiumej.v27i2.4124

4.2. Qualitative Results and Visual Comparison

We compare the performance of our PromptLessSAM model with several other well-known methods for crack segmentation. We chose these methods for comparison for specific reasons. We compare against the U-Net [4] as a baseline CNN architecture. We include Deepcrack [7] and TransUNet [6] because they are popular, high-performance models, but they have a very large number of trainable parameters (14.72M and 101.19M, respectively). We also compare against SAM-based methods like PaveSAM [9] and CrackESS [10] because they also use the SAM foundation model, but in a different way.

Table 2 shows the quantitative results on the Crack500 dataset. PromptLessSAM achieves a high Dice score of 72.7% and an IoU of 57.11%, which outperforms U-Net [4], Deepcrack [7] and TransUNet [6,18]. It also surpasses the SAM-based methods (SAM [8], PaveSAM [9], and CrackESS [10]), demonstrating that prompt-less adaptation can be effective. In addition, we include more recent crack segmentation networks with lower reported results on Crack500, such as UP-CrackNet [16] and WP-CrackNet [17], as well as SwinCrack [18,21], VM-UNet32 [18,21], and DTrC-Net [19], to provide a broader comparison. Overall, PromptLessSAM remains the best-performing model in terms of recall and Dice score.

The other important result is the comparison of performance and model size. PromptLessSAM achieves the best segmentation performance while having the fewest trainable parameters (1.1M). This has almost 100 times fewer parameters than TransUNet [6,18] (101.19M) and 13 times fewer than Deepcrack [7] (14.72M). This shows that our method is extremely training-efficient.

Table 2. Evaluation metrics of competing methods on the Crack500 dataset.

Methods	Precision [%]	Recall [%]	F1 [%]	IoU [%]	Trainable Params [M]
U-Net [4]	63.90	68.40	65.10	–	13.34
Deepcrack [7]	62.80	66.10	67.60	–	14.72
TransUNet [6,18]	60.50	69.60	64.70	–	101.19
SwinCrack [18,20]	**73.38**	68.26	70.73	**72.77**	116.23
VM-UNet32 [18,21]	70.27	72.35	71.69	–	27
DTrC-Net [19]	–	–	67.50	53.30	41.8
UP-CrackNet [16]	65.38	58.61	61.81	44.73	–
WP-CrackNet [17]	71.81	55.58	62.66	45.63	–
CrackESS (4-point) [10]	51.70	70.50	56.10	40.70	–
SAM [9]	10.50	67.20	14.80	9.50	–
PaveSAM (Dice + BCE loss) [9]	69.60	72.30	69.10	53.80	3.87
PromptLessSAM (Ours)	70.25	**75.32**	**72.70**	57.11	**1.09**

A graph was used to demonstrate the quantitative comparison of models' performance based on the evaluation metrics mentioned above. Figure 4 shows that our model achieved the highest recall (75.32%) and the highest Dice score (72.70%) among state-of-the-art models. At the same time, SwinCrack [18,20] achieved the highest precision and IoU among the reported results, indicating strong overlap performance with a large model. However, when considering the trainable parameters in Table 2, PromptLessSAM provides a better trade-off between segmentation accuracy and training cost. This confirms that lightweight decoder adaptation can still achieve competitive crack segmentation without relying on prompts.

https://doi.org/10.31436/iiumej.v27i2.4124

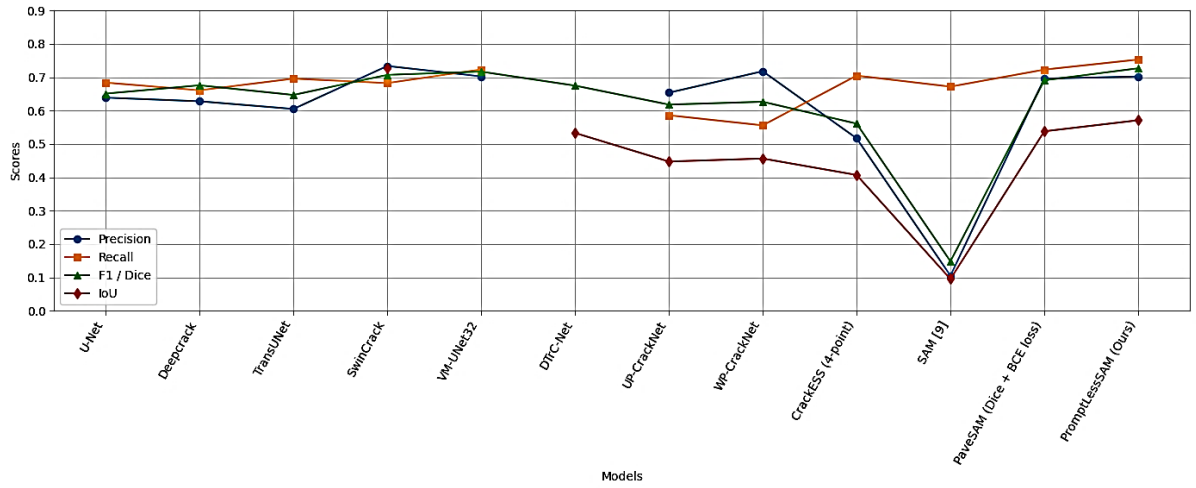
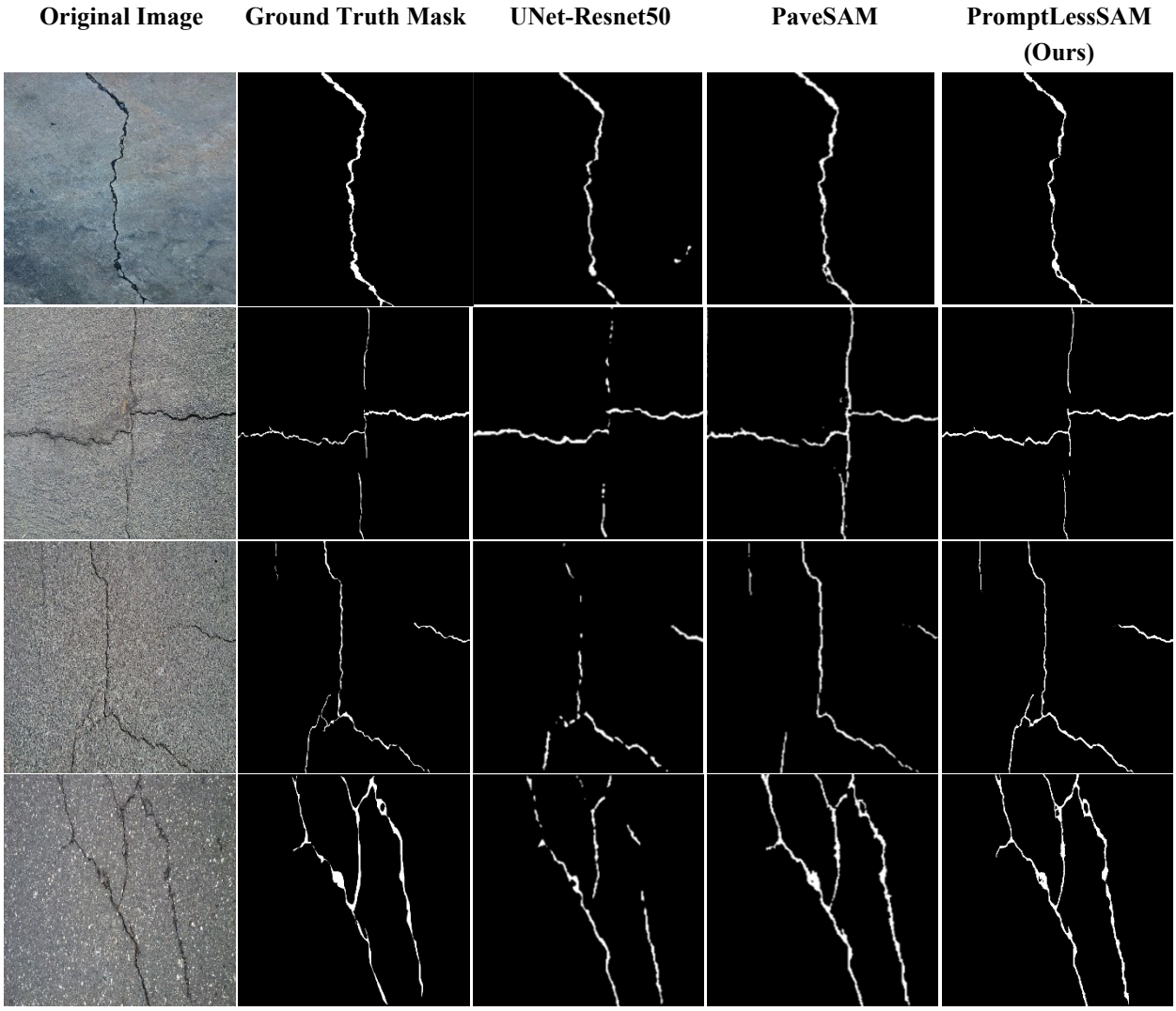


Figure 4. Quantitative comparison of performance metrics across different segmentation models.

Visual examples of our model's output on the Crack500 dataset (test data) are illustrated in Figure 5.



https://doi.org/10.31436/iiumej.v27i2.4124

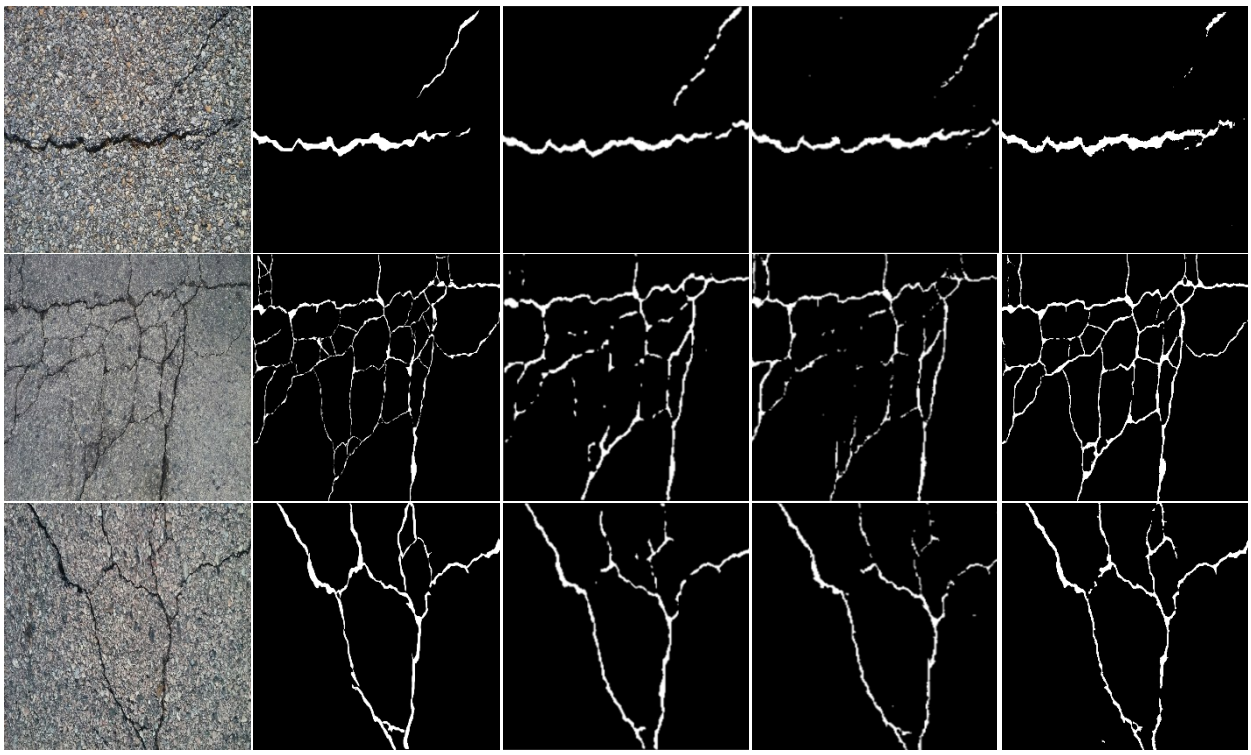


Figure 5. Visual comparison of segmentation results on the Crack500 test set.

The visual results demonstrate that our model generalizes well to different crack shapes and backgrounds, such as asphalt and concrete, in the dataset.

4.3. Generalization on DeepCrack Dataset

To check if our model simply memorized the Crack500 dataset, we tested its generalization ability. We took our trained PromptLessSAM model (which was trained only trained on Crack500) and tested it directly on the DeepCrack dataset [7]. The model had never seen any images from this dataset before.

The quantitative evaluation on this unseen dataset demonstrates the robust performance of our method. Specifically, PromptLessSAM achieved a precision of 79.32% and a recall of 78.25%. Furthermore, the model achieved a strong F1 Score of 74.70% and an IoU of 62.43%. This performance is slightly better than its F1 score on the Crack500 dataset (73.3%). This is an important result because it shows that the model is not overfitting to the training data. This result demonstrates that our fine-tuning strategy has learned a robust, general representation of what a “crack” is and can be applied to new, unseen datasets. This proves that our method is a reliable and general-purpose solution for crack segmentation. Visual examples of our model’s output on the DeepCrack dataset are shown in Figure 6.

https://doi.org/10.31436/iiumej.v27i2.4124

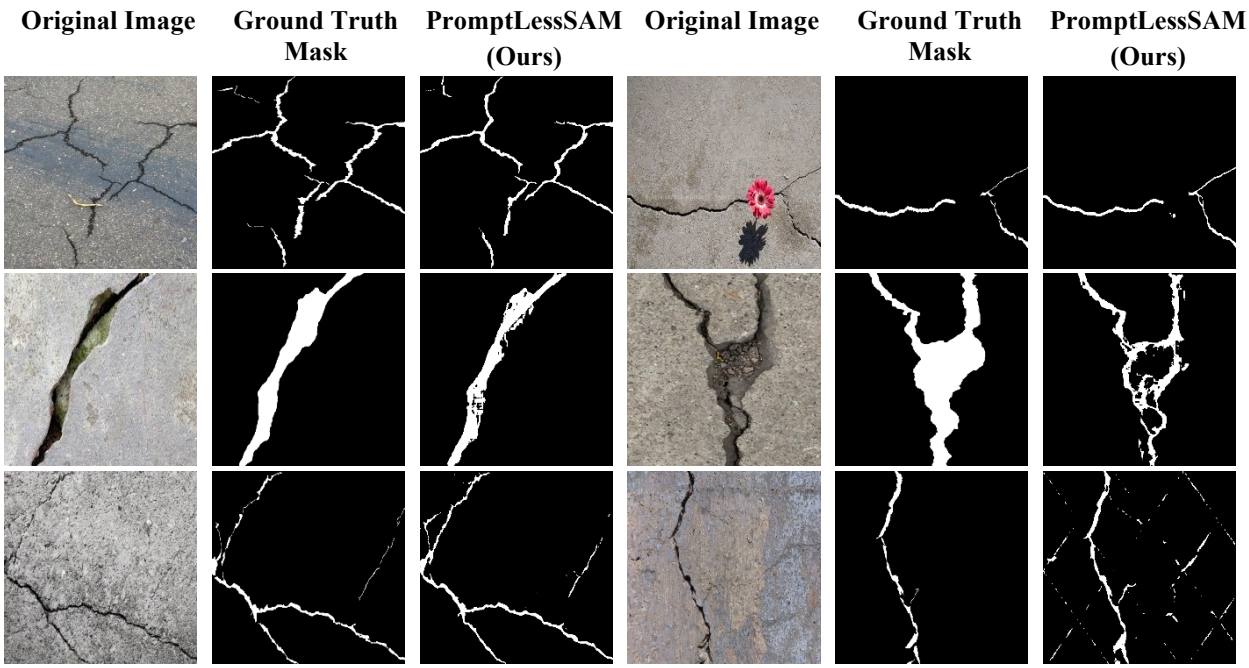


Figure 6. Visual examples of our model's generalization performance on the DeepCrack dataset (never seen).

5. DISCUSSION

PromptLessSAM demonstrates that a large pre-trained model can be effectively repurposed for a specialized task with minimal training. By removing the dependency on prompts, we simplify deployment; an inspector using our model does not need to provide any hint; the model directly outputs crack locations. This contrasts with PaveSAM, which requires drawing a box around each distress [9]. Our approach can thus be integrated into autonomous inspection systems, such as those mounted on drones or robots, to enable fully automated crack mapping.

The Dice score of 72.7% on Crack500 should be interpreted in the context of dataset difficulty. Crack500 includes challenging imaging conditions such as shadows, occlusions, varying lighting, and strong background interference, which makes accurate segmentation of thin cracks harder [17,18]. For this reason, many published methods on Crack500 commonly report Dice/F1 values in the 60% to low 70% range (for example, WP-CrackNet [17] and UP-CrackNet [16] are around 62%, while SwinCrack [20] and VM-UNet32 [18] are around 71%). However, on other crack datasets with different characteristics, several models can reach close to 80% F1, such as the reported results on Crack756 in recent Mamba-based studies [18]. Therefore, our Crack500 result indicates that the proposed prompt-less SAM adaptation handles a hard benchmark while keeping the adaptation lightweight.

In addition, our implementation was initially based on an open-source promptless task-specific fine-tuning pipeline [29] built on top of the SAM framework. This reference work provides a practical baseline for adapting SAM to a single target class without interactive prompting, aligning with our goal of fully automatic crack segmentation. However, our final model differs in several important aspects to better fit the crack segmentation problem, especially by changing the loss functions and tuning various training parameters to address the severe class imbalance between crack pixels and background. Therefore, while we acknowledge this codebase as a starting point, our modifications are essential to improve performance and to ensure the contribution of our proposed method to pavement crack datasets.

One limitation is that SAM's encoder operates at a fixed input size (1024×1024). For very large images, one must tile or resize, which could affect very small cracks. Another point is inference speed: while adequate models achieve much higher speed, there is a trade-off between segmentation quality and speed. For real-time requirements, one might opt for a smaller SAM or distill our model.

In terms of broader impact, our work reinforces the idea that foundation models can serve as powerful backbones for domain-specific tasks in civil engineering. The success of PromptLessSAM suggests that training task-specific models from scratch might become unnecessary when such pre-trained models are available. This can significantly reduce development time and the amount of required training data for new vision tasks in SHM and beyond.

6. CONCLUSION

In this paper, we propose PromptLessSAM, an efficient method for adapting the large SAM foundation model to the specific task of pavement crack segmentation. We addressed the main problem of existing methods: the poor domain-specific performance of generalist models like SAM. Our method freezes the powerful SAM ViT-H backbone, fine-tunes the encoder's Neck, and trains a new lightweight decoder. This adaptation makes the model "prompt-less" and fully automatic.

Our experiments on the Crack500 dataset show that our model is extremely efficient, with only 1,093,345 (~ 1.1 million) trainable parameters. With this lightweight design, our model achieves a Dice score of 72.7%. This performance is higher than that of many other popular models. Furthermore, our model showed good generalization, achieving a 74.7% Dice score on the unseen DeepCrack dataset. This work shows that this lightweight adaptation strategy is highly effective and efficient for turning foundation models into domain experts.

For future work, several clear directions are planned. First, we aim to improve the Dice similarity coefficient by using stronger training protocols and greater data diversity, including additional augmentations, longer training schedules, and patch-based training and inference to refine thin-crack boundaries and improve continuity. Second, deployment aspects should also be addressed by reducing encoder cost through model compression, including quantization. Finally, acceleration and optimized export for mobile and edge devices will be explored, following systems such as CrackESS.

REFERENCES

[1] Sundravel KV, Jagadeesan R, Dhanush A, Kumar AS (2025). A Review of Innovations, Challenges, and Future Directions in Structural Health Monitoring Using Smart Materials. Institute for Environmental Nanotechnology. doi:10.13074/jent.2025.03.2511163.

[2] Zhang X, Wang H, Hsieh YA, Yang Z, Yezzi A, Tsai YC (2025). Deep Learning for Crack Detection: A Review of Learning Paradigms, Generalizability, and Datasets. Retrieved from http://arxiv.org/abs/2508.10256

[3] Benmhahe B, Chentoufi JA (2021). Automated Pavement Distress Detection, Classification and Measurement: A Review. International Journal of Advanced Computer Science and Applications, 12(8):708–718. https://doi.org/10.14569/IJACSA.2021.0120882

[4] Ronneberger O, Fischer P, Brox T (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp 234–241. Retrieved from http://arxiv.org/abs/1505.04597

[5] Chen X, Liu C, Chen L, Zhu X, Zhang Y, Wang C (2024). A Pavement Crack Detection and Evaluation Framework for a UAV Inspection System Based on Deep Learning. Applied Sciences, 14(3). doi:10.3390/app14031157.

[6] Chen J, Lu Y, Yu Q, Luo X, Adeli E, Wang Y, Lu L, Yuille AL, Zhou Y (2021). TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation. Retrieved from https://arxiv.org/pdf/2102.04306

[7] Liu Y, Yao J, Lu X, Xie R, Li L (2019). DeepCrack: A deep hierarchical feature learning architecture for crack segmentation. Neurocomputing, 338:139–153. doi:10.1016/j.neucom.2019.01.036.

[8] Kirillov A, Mintun E, Ravi N, Mao H, Rolland C, Gustafson L, Xiao T, Whitehead S, Berg AC, Lo WY, Dollár P, Girshick R (2023). Segment Anything. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp 4015-4026. Retrieved from http://arxiv.org/abs/2304.02643

[9] Jakisa N, Student OP, Adu-Gyamfi Y, Aboah A, Amo-Boateng M (2024). PaveSAM-Segment Anything for Pavement Distress.

[10] Wang Y, He J, Yu S (2025). CrackESS: A Self-Prompting Crack Segmentation System for Edge Devices. Retrieved from http://arxiv.org/abs/2412.07205

[11] Zhang D, Feng T, Xue L, Wang Y, Dong Y, Tang J (2025). Parameter-Efficient Fine-Tuning for Foundation Models. Retrieved from http://arxiv.org/abs/2501.13787

[12] Huang S, Chen H, Yan L, Zou X, Li B, Bi Y (2025). A review of the progress in machine vision-based crack detection and identification technology for asphalt pavements. Digital Transportation and Safety, 4(1):65–79. doi:10.48130/dts-0025-0006.

[13] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In International Conference on Learning Representations (ICLR). Retrieved from http://arxiv.org/abs/2010.11929

[14] Li X, Ding H, Yuan H, Zhang W, Pang J, Cheng G, Chen K, Liu Z, Loy CC (2024). Transformer-Based Visual Segmentation: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 46(12):10138-10159. Retrieved from http://arxiv.org/abs/2304.09854

[15] Zhao X, Ding W, An Y, Du Y, Yu T, Li M, Tang M, Wang J (2023). Fast Segment Anything. Retrieved from http://arxiv.org/abs/2306.12156

[16] Ma N, Fan R, Xie L. (2024). UP-CrackNet: Unsupervised Pixel-Wise Road Crack Detection via Adversarial Image Restoration. IEEE Transactions on Intelligent Transportation Systems, 25(10):13926–13936. doi: 10.1109/TITS.2024.3398037

[17] Ma N, Song Z, Hu Q, Tang X, Zhang C, Fan R, Xie L. (2026). WP-CrackNet: A collaborative adversarial learning framework for end-to-end weakly-supervised road crack detection. Neurocomputing, 659:131845. doi: 10.1016/j.neucom.2025.131845

[18] Wang Y, Jin J, Chen X, Wu Z, Zhang L. (2025). A lightweight crack segmentation network based on the importance-enhanced Mamba model. Scientific Reports, 15(1):25504. doi: 10.1038/s41598-025-25504-4

[19] Tang W, Wu Z, Wang W, Pan Y, Gan W. (2025). VM-UNet++ research on crack image segmentation based on improved VM-UNet. Scientific Reports, 15(1):92994. doi: 10.1038/s41598-025-92994-7

[20] Wang C, Liu H, An X, Gong Z, Deng F. (2024). SwinCrack: Pavement crack detection using convolutional swin-transformer network. Digital Signal Processing, 145:104297. doi: 10.1016/j.dsp.2023.104297

[21] Ruan J, Li J, Xiang S. (2025). VM-UNet: Vision Mamba UNet for Medical Image Segmentation. ACM Transactions on Multimedia Computing, Communications, and Applications, 20(23):1-15. doi: 10.1145/3767748

[22] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. Retrieved from http://arxiv.org/abs/1912.01703

[23] Kingma DP, Ba J (2017). Adam: A Method for Stochastic Optimization. Retrieved from http://arxiv.org/abs/1412.6980

[24] Jadon S (2020). A survey of loss functions for semantic segmentation. In IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB). doi:10.1109/CIBCB48159.2020.9277638.

[25] Rajput V (2021). Robustness of different loss functions and their impact on network's learning capability. Retrieved from https://arxiv.org/abs/2110.08322

[26] Hussan PH, Ali IH (2025). ECGANCOVID: Efficient Conditional GAN Architecture for Covid-19 Disease Segmentation. Baghdad Science Journal, 22(2):706–729. doi:10.21123/bsj.2024.9335.

[27] Al Akabi H, Al-Assadi T (2025). CLASSIFICATION OF FAULT SIGNALS BASED ON DCT AND DEEP LEARNING. Kufa Journal of Engineering, 16(4):217–234. doi:10.30572/2018/KJE/160413.

[28] Runpod. Runpod | The cloud built for AI. Retrieved from https://www.runpod.io/

[29] Yatnalkar Y (2025). SAM-Promptless-Task-Specific-Finetuning. Retrieved from https://github.com/yogendra-yatnalkar/SAM-Promptless-Task-Specific-Finetuning