Highlights

SAM-LLaVA: A Segmentation-Aware Vision-Language Framework for Industrial Defect Diagnosis

Shengwang An, Chengjia Wang, Xinghui Dong

- Proposing a CLIP-SAM cascade prompting mechanism that leverages CLIP's zero-shot generalization for coarse defect localization and guides SAM for pixel-precise segmentation, overcoming the limitation of fixed patch-level similarity computation for micro-scale anomalies.
- Designing the Segmentation-Aware Semantic Alignment (SASA) module to establish bidirectional cross-modal attention between pixel-level segmentation masks and textual embeddings, significantly reducing hallucinations and enhancing consistency in defect descriptions.
- Introducing a multi-scale prompt learner and mask decoder architecture to adaptively handle industrial defects of varying scales, from microscopic scratches to large-area stains, improving model robustness across diverse manufacturing scenarios.

SAM-LLaVA: A Segmentation-Aware Vision-Language Framework for Industrial Defect Diagnosis

Shengwang An[a], Chengjia Wang[b], Xinghui Dong[a,*]

[a]*The State Key Laboratory of Physical Oceanography and the Faculty of Information Science and Engineering, Ocean University of China, Qingdao, 266100, China*
[b]*School of Mathematical and Computer Sciences, Heriot-Watt University, Riccarton Mains Road, EH14 4AS, Edinburgh, United Kingdom*

Abstract

Large Vision-Language Models (LVLMs) have demonstrated potential in industrial defect diagnosis. However, their reliance on patch-level similarity computation and generic textual priors limits their capability for fine-grained defect localization and semantic alignment, often resulting in imprecise detection and hallucinated descriptions. Existing methods typically adopt a "one-class-one-model" paradigm or use global feature matching, which limits their adaptability to novel defects and precise defect description. To address these issues, we propose SAM-LLaVA, a segmentation-aware vision-language framework for zero-shot and few-shot industrial defect diagnosis. Our method introduces three key innovations: (1) a CLIP-SAM cascade prompting mechanism that leverages CLIP's zero-shot generalization for coarse localization and guides Segment Anything Model (SAM) for pixel-precise segmentation; (2) a multi-scale prompt learner and mask decoder that enhances adaptability to defects of varying sizes; and (3) a Segmentation-Aware Semantic Alignment (SASA) module that establishes bidirectional cross-modal alignment between segmentation masks and textual embeddings, reducing hallucination and improving description consistency. Extensive experiments on MVTec-AD, VisA, and our Text-Augmented Defect Data Set (TADD) demonstrate that SAM-LLaVA achieves state-of-the-art performance in both defect de-

⋆Source code and models will be made publicly available upon acceptance of the paper.

*Corresponding author.
Email addresses: 21230211077@stu.ouc.edu.cn (Shengwang An), chengjia.wang@hw.ac.uk (Chengjia Wang), xinghui.dong@ouc.edu.cn (Xinghui Dong)

tection and textual description under zero-shot and few-shot settings. Ablation studies confirm the contribution of each component, highlighting the effectiveness of our integrated coarse-to-fine and segmentation-aware design.

Keywords: Industrial Defect Detection; Large Vision-Language Models; Zero-Shot Learning; Few-Shot Learning; Semantic Segmentation

---

1. Introduction

Industrial Defect Detection (IDD) plays a critical role in ensuring manufacturing quality and product reliability. Traditional methods often rely on handcrafted features and classical image processing techniques, which struggle with complex or subtle defects due to limited generalization ability. In recent years, deep learning-based approaches [1, 2] have emerged as dominant solutions, leveraging Convolutional Neural Networks (CNNs) [3] or Vision Transformers (ViTs) [4] to learn discriminative features from raw data. Methods such as PatchCore [5] and SPADE [6] achieve strong performance through memory bank construction or feature pyramid matching. However, these approaches [7, 8] typically follow a "one-class-one-model" paradigm, requiring substantial labeled data and exhibiting limited adaptability to novel defect categories, thereby hindering their practical deployment in dynamic industrial environments.

In addition, changes in product type, surface material, or lighting conditions often necessitate complete model retraining from scratch, leading to substantial downtime and annotation costs that are impractical for most industrial settings. Even within the same production line, defect appearances may vary significantly across batches due to seasonal raw-material fluctuations or machine aging, further amplifying the data hunger of conventional supervised frameworks. Consequently, the industry urgently demands unified solutions that can be rolled out to multiple product lines with minimal manual intervention.

With the advancement of Large Vision-Language Models (LVLMs), researchers have begun to explore their application in defect detection tasks, aiming to enhance semantic understanding through linguistic priors. Existing methods [9, 10] leverage CLIP [11] for cross-modal alignment in zero-shot or few-shot settings, while recent efforts

such as SPGDD-GPT [12] further integrate text-driven defect focusing and multi-scale memory mechanisms to improve generalization and description quality. Despite these advances, existing LVLM-based methods [13] still face several key challenges.

First, their visual encoders often rely on fixed patch-level similarity computations, which limit precise localization of defects, especially for micro-scale anomalies that occupy only a few pixels. Second, when feeding detection results into Large Language Models (LLMs), these methods [14, 15] typically concatenate image features and text tokens without establishing deep semantic alignment, leading to inconsistent or hallucinated descriptions that do not accurately reflect visual evidence. Third, most frameworks treat defect localization and description as two isolated stages, in which the segmentation mask (if any) has never been revisited by the language decoder. As a result, spatial details such as the exact boundary, shape irregularity, or relative position to key product features are often omitted or misrepresented. Consequently, inspectors still need to manually verify the reported locations and descriptions, defeating the original purpose of automatic diagnosis and preventing closed-loop quality control.

Recently, An et al. [12] introduced a Text-Driven Defect Focuser (TDDF) and a Multi-scale Self-prompted Memory Module (MSSPMM) for SPGDD-GPT, to improve few-shot generalization and description quality. Although these advances strengthen feature-space alignment, three fundamental limitations remain unresolved. First, the detection granularity is still bounded by the fixed patch-level similarity of CLIP [11], making micro-scale anomalies difficult to localize. Second, defect localization and description are treated as two isolated stages. In this case, the segmentation mask (if any) has never been revisited by the language decoder, causing spatial details, such as boundaries and shapes, to be omitted or hallucinated. Third, the framework lacks an explicit pixel-level segmentation mechanism to bridge visual evidence and linguistic concepts. These limitations motivate the paradigm shift that we explore in this study.

To address the aforementioned limitations, we propose SAM-LLaVA, a segmentation-aware vision-language framework that fundamentally departs from the end-to-end feature alignment paradigm of SPGDD-GPT. Unlike SPGDD-GPT, which operates solely in the CLIP feature space through both TDDF and MSSPMM, SAM-LLaVA introduces an intermediate pixel-level segmentation stage via a CLIP-SAM cascade, and explic-

itly injects spatial priors into the language decoder via a novel Segmentation-Aware Semantic Alignment (SASA) module. This design transforms the task from SPGDD-GPT's "detect-then-describe" paradigm into our "segment-align-describe" paradigm, enabling pixel-precise localization and hallucination-free description. To be specific, our SAM-LLaVA integrates three core stages, including (1) Coarse Localization with CLIP, which generates initial anomaly heatmaps using contrastive text-image alignment; (2) Fine Segmentation with Segment Anything Model (SAM) [16], where coarse masks serve as geometric prompts to guide the SAM toward pixel-precise anomaly segmentation; and (3) Segmentation-Aware Description with LLaVA, in which the SASA module aligns pixel-level segmentation maps with textual embeddings before feeding them into LLaVA [17] for detailed, consistent defect description.

We also introduce a Multi-scale Prompt and Mask Decoder to handle defects of varying scales and a Contextual Similarity Adaptation mechanism for robust few-shot learning. By explicitly injecting spatial prior into the language decoder, SASA significantly reduces hallucination and ensures that the generated text faithfully corresponds to the exact defect region, thus providing trustworthy reports that can be directly used for downstream repair or quality grading without further human inspection. Furthermore, the entire pipeline is optimized end-to-end with a lightweight LoRA [18] fine-tuning strategy, enabling rapid deployment on edge devices with limited GPU memory. Extensive experiments on MVTec-AD [19], VisA [20], and our newly collected TADD [12] benchmarks demonstrate that SAM-LLaVA consistently outperforms state-of-the-art competitors in both detection accuracy and description fidelity, highlighting its practical value in real-world production lines.

Our contributions can be summarized as threefold.

1. A coarse-to-fine detection pipeline that combines CLIP zero-shot generalization with SAM's precise segmentation capability, eliminating the need for manual threshold tuning and improving localization accuracy, especially for fine-grained defects.
2. A segmentation-aware semantic alignment module that establishes bidirectional cross-attention between image features and text embeddings, reducing descrip-

tion hallucinations and enhancing consistency between detected regions and generated text.

3. A multi-scale prompt learning and mask decoding architecture that adapts to defects of varying sizes, from microscopic scratches to large-area stains, improving model robustness across diverse industrial scenarios.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 details the proposed SAM-LLaVA framework. The experimental setup is described in Section 4. Section 5 presents the results of the quantitative and qualitative evaluation and ablation study. Finally, Section 6 concludes the paper.

2. Related Work

2.1. Defect Detection

Traditional defect detection methods [21, 22] typically rely on handcrafted features or classical image processing techniques, which often struggle with complex or subtle defects due to limited generalization. Deep learning-based approaches have since dominated the field, with methods such as PatchCore [5] and SPADE [6] achieving strong performance through memory bank construction or feature pyramid matching. However, these methods are generally trained under a "one-class-one-model" paradigm, requiring substantial labeled data and lacking adaptability to novel defect categories.

Recent advances in foundation models have introduced new paradigms for defect analysis. WinCLIP [23] and AnomalyGPT [14] leverage cross-modal alignment of CLIP [11] for zero-shot or few-shot defect detection, but their reliance on global feature similarity limits localization precision. On the detection front, SAM-based methods such as SAA [24] and SAID [25] generate candidate masks from textual prompts but suffer from redundant outputs and ambiguous localization. ClipSAM [26] attempted to integrate CLIP and SAM into a two-stage pipeline, using CLIP for rough detection, then passing the rough detection to SAM for refinement. This strategy motivated our work, but due to the possible bias in CLIP's rough detection, directly feeding SAM may bring noise to SAM. Therefore, we only use CLIP's rough detection as a hint for SAM to reduce noise and achieve better positioning detection results. Recent studies,

such as [27, 28], have also explored SAM enhanced anomaly detection framework, but they still lack close integration with visual language semantic alignment.

2.2. Zero-Shot and Few-Shot Defect Detection

Zero-Shot Defect Detection (ZSDD) normally transfers a model which has been trained in a source domain to a target domain without annotated data. Existing ZSDD methods [23, 14], typically employ a pre-trained CLIP [11] model to compare the similarity between test images and textual descriptions of normal and abnormal conditions in the feature space. However, these methods often struggle with generalizability in real-world scenarios where domain shifts occur, and they provide only anomaly scores that require manual threshold tuning.

On the other hand, Few-Shot Defect Detection (FSDD) recognizes defects with only a few normal samples contained in the target domain. For example, InCTRL [9] evaluated residuals between query images and a small set of normal sample prompts, while RegAD [8] used image registration networks for alignment. More recently, AnomalyGPT [14] leveraged LVLMs to detect defects using a few normal samples for contextual learning, and SPGDD-GPT [12] further enhanced few-shot capability through its MSSPMM module. Although these methods improve the adaptability of limited samples, they still face a key limitation, i.e., detection granularity, which relies on the patch-level similarity of CLIP [11]. As a result, performing accurate defect detection is challenging.

In contrast, our method introduces a coarse-to-fine detection pipeline where CLIP [11] provides initial localization cues to guide SAM [16] precise detection, eliminating the need for manual threshold selection in zero-shot settings. For few-shot learning, we propose a contextual similarity adaptation module that leverages CLIP [11] coarse detection to compute similarity with normal samples, enabling more robust adaptation. Furthermore, our model generates detailed, Segmentation-Aware textual descriptions by integrating LLaVA [17] with Segmentation-Aware alignment.

2.3. Vision-Language Models for Industrial Diagnosis

Large Vision-Language Models (LVLMs), such as Qwen3-VL [29], LLaVA [17], and PandaGPT [30], have shown impressive capabilities in visual understanding and

conversational response generation. In the industrial domain, AnomalyGPT [14] pioneered the use of LVLMs for defect detection and description, employing prompt tuning to adapt a pre-trained Vicuna [31] model. SPGDD-GPT [12] further enhanced this line by introducing a Text-Driven Defect Focuser (TDDF) and a Multi-scale Self-prompted Memory Module (MSSPMM) to improve few-shot generalization and defect description quality. However, SPGDD-GPT remains an end-to-end feature alignment framework. Specifically, it relies on the patch-level similarity of CLIP [11] for localization and concatenates image features with text tokens for description generation. Consequently, SPGDD-GPT cannot produce pixel-precise segmentation masks and cannot establish explicit bidirectional alignment between pixel-level spatial layouts and word-level semantics.

In contrast, SAM-LLaVA addresses these gaps by introducing a physical segmentation stage (via SAM [16]) and a bidirectional cross-attention module (SASA) that operates directly on pixel-mask embeddings, rather than on coarse feature maps. As a result, SAM-LLaVA realizes a paradigm shift from "feature-space defect focusing" to "pixel-level segmentation-aware diagnosis".

3. Methodology

We propose SAM-LLaVA, a coarse-to-fine vision-language framework for zero-shot and few-shot industrial defect diagnosis. As illustrated in Fig. 1, our method consists of three main stages: (1) Coarse Localization with CLIP [11], where we extract patch-wise anomaly scores and generate a rough segmentation mask; (2) Fine Segmentation with SAM [16], where the coarse mask is used as a spatial prompt to guide SAM in producing pixel-precise anomaly segmentation; and (3) Segmentation-Aware Description with LLaVA [17], where the refined segmentation map and original image are aligned with textual embeddings via a novel Segmentation-Aware Semantic Alignment (SASA) module, then fed into LLaVA for detailed defect description. Additionally, we introduce a Multi-scale Prompt and Mask Decoder to handle defects of varying scales, and a Contextual Similarity Adaptation mechanism for few-shot learning.

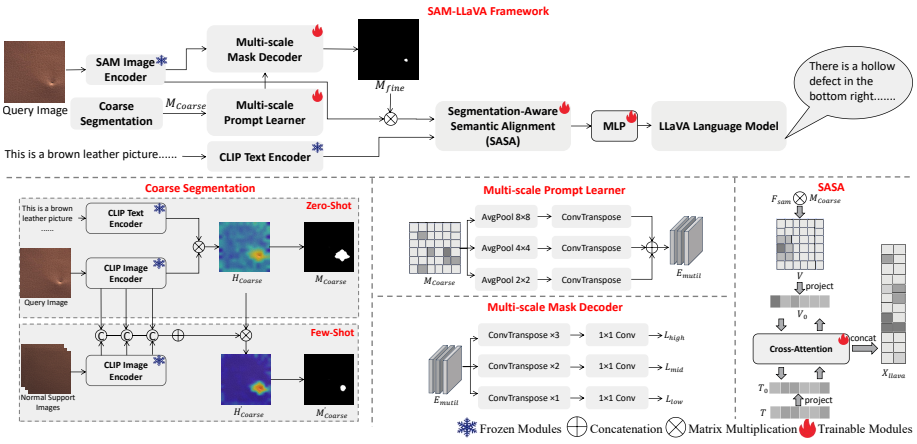

Figure 1: The overall architecture of the proposed SAM-LLaVA framework. Under the support of zero- or few-shot, the query image first generates a fine defect mask through the CLIP-SAM cascade module. Subsequently, the segmentation mask interacts with the image features through the segmentation aware semantic alignment module, and is finally fed into LLaVA [17] to generate detailed defect descriptions.

3.1. CLIP-SAM Coarse-Fine Cascaded Prompting

This module addresses a fundamental limitation of SPGDD-GPT [12] and other CLIP-based methods. To be exact, their reliance on fixed patch-level similarity computation limits precise localization of micro-scale anomalies. Although the TDDF of SPGDD-GPT attempts to improve feature-space discrimination, it cannot overcome CLIP's inherent patch granularity. One of our core innovations is to leverage CLIP's zero-shot generalization for coarse localization, then convert the abstract anomaly score into a physical spatial prompt (coarse mask) for SAM [16], enabling pixel-level fine segmentation which is impossible for pure feature alignment methods.

Under the zero-shot setting, we utilize the frozen image encoder $f_{\text{clip}}^{\text{img}}$ and text encoder $f_{\text{clip}}^{\text{txt}}$ of CLIP. By computing the contrastive distance between the image features and pre-defined normal/abnormal text embeddings, a coarse-grained defect response heatmap $H_{\text{coarse}} \in \mathbb{R}^{H' \times W'}$ is generated. This heatmap is binarized using an adaptive threshold τ to obtain the initial coarse mask prompt:

$$M_{\text{coarse}} = \mathbb{I}(H_{\text{coarse}} > \tau) \in \{0, 1\}^{H' \times W'}, \tag{1}$$

where τ is automatically determined by Otsu's method [32] to maximize the separabil-

ity between the foreground and background intensities.

Using the few-shot setting, we introduce K normal support samples $\mathcal{D}_{\text{support}}$ to calibrate the above coarse heatmap. First, the normal support example is sent to the image encoder of CLIP, and multi-level image features are extracted from the 3, 8 and 12 layers of the encoder. These features are used to build a multi-scale normal memory $\{\mathcal{M}^s\}_{s=1}^{S}$ (where $S = 3$ corresponds to the three layers). For the query image, we compute the minimum cosine distance map d_s between its features at each layer s and the nearest neighbor in the corresponding memory bank. These multi-scale distance maps are spliced into a context-aware anomaly score map s_{ctx}. This score map is transformed into a modulation coefficient map Λ via a Sigmoid function, which is used to suppress (normal regions) or enhance (defective regions) the initial coarse heatmap:

$$H'_{\text{coarse}} = H_{\text{coarse}} \odot \Lambda. \tag{2}$$

The calibrated heatmap H'_{coarse} is then thresholded to yield a more precise coarse mask M'_{coarse}.

Subsequently, the coarse mask M_{coarse} (zero-shot) or M'_{coarse} (few-shot) is used as a mask prompt, along with the query image I_q, and fed into SAM. Let SAM image encoder, prompt encoder, and mask decoder be $f_{\text{sam}}^{\text{img}}$, $f_{\text{sam}}^{\text{prompt}}$, and $f_{\text{sam}}^{\text{mask}}$, respectively. The encoding process can be expressed as:

$$\mathbf{F}_{\text{sam}} = f_{\text{sam}}^{\text{img}}(I_q), \quad \mathbf{E}_{\text{prompt}} = f_{\text{sam}}^{\text{prompt}}(M_{\text{coarse}}). \tag{3}$$

The mask decoder receives the concatenated features and outputs candidate masks along with their confidence scores:

$$\mathbf{M}_{\text{logits}}, \mathbf{s}_{\text{iou}} = f_{\text{sam}}^{\text{mask}}(\mathbf{F}_{\text{sam}} + \mathbf{E}_{\text{prompt}}), \tag{4}$$

where $\mathbf{M}_{\text{logits}} \in \mathbb{R}^{N_m \times H_o \times W_o}$ denotes the logits for N_m candidate masks, and $\mathbf{s}_{\text{iou}} \in \mathbb{R}^{N_m}$ represents the predicted IoU scores. We select the mask with the highest confidence, upsample it, and binarize it to obtain the final fine segmentation mask $M_{\text{fine}} \in \{0, 1\}^{H \times W}$.

3.2. Multi-scale Prompt Learner and Mask Decoder

To enhance the model adaptability to defects of varying scales (from microscopic scratches to large-area stains), we introduce multi-scale designs in both the prompt en-

coding and mask decoding stages of SAM. Given the coarse mask $M_{\text{coarse}} \in \{0,1\}^{H\times W}$, we construct three parallel branches to capture multi-scale defect contexts. Each branch applies average pooling with kernel size $k \in \{8,4,2\}$ and stride m, followed by a transposed convolution ($k \times k$, stride m) to restore the resolution to $h \times w$ pixels, producing $\{G_{\text{large}}, G_{\text{medium}}, G_{\text{small}}\} \subset \mathbb{R}^{h\times w\times C}$. Each G is then encoded by an independent 1×1 convolution (number of channels output is C), and the results are concatenated to form the multi-scale prompt:

$$E_{\text{multi}} = [\text{Conv}_{1\times1}(G_{\text{large}}); \text{Conv}_{1\times1}(G_{\text{medium}}); \text{Conv}_{1\times1}(G_{\text{small}})] \in \mathbb{R}^{h\times w\times 3C}. \quad (5)$$

The Multi-scale Mask Decoder extends the original SAM decoder, incorporating parallel prediction branches at the high (L_{high}), medium (L_{mid}), and low (L_{low}) resolutions. During the training stage, all branches are supervised. The total segmentation loss is a weighted sum of the losses from each branch:

$$\mathcal{L}_{\text{seg}} = \sum_{s\in\{\text{high,mid,low}\}} \omega_s \cdot \Big(\mathcal{L}_{\text{dice}}(M^s_{\text{gt}}, \sigma(L_s)) + \lambda\mathcal{L}_{\text{focal}}(M^s_{\text{gt}}, \sigma(L_s))\Big), \quad (6)$$

where M^s_{gt} is the ground-truth mask downsampled to the resolution of branch s, $\lambda = 1.0$ is a balancing coefficient, and the weights ω_s are empirically set to $\omega_{\text{high}} = 0.3, \omega_{\text{mid}} = 1.0, \omega_{\text{low}} = 0.2$ to emphasize the optimization of the medium-resolution branch used for final inference. During the inference stage, for efficiency, only the output from the medium-resolution branch L_{mid} is used to generate the final mask M_{fine}.

3.3. Segmentation-Aware Semantic Alignment Module

To establish a precise correspondence between pixel-level segmentation results and word-level semantic descriptions, thereby suppressing description hallucinations, we insert a lightweight bidirectional cross-attention module, namely, Segmentation-Aware Semantic Alignment (SASA) module, before the image features are fed into LLaVA [17]. This module takes three inputs, including the fine segmentation mask M_{fine}, the SAM image features $\mathbf{F}_{\text{sam}}$, and the CLIP text embeddings $\mathbf{T}_{\text{clip}}$.

The SASA module fundamentally differs from the TDDF of SPGDD-GPT, which optimizes the Euclidean distance between image features and text embeddings to "focus" attention in the feature space. Although TDDF is unidirectional and operates at the

feature level, SASA performs bidirectional cross-attention between $\mathbf{V}'$ (mask-grounded image features at specific pixel coordinates) and $\mathbf{T}'$ (text embeddings), ensuring that each generated word is directly anchored to specific spatial locations in M_{fine}.

The image features are first multiplied pixel by pixel by the mask to obtain defect-aware visual features $\mathbf{V}$, which are then projected, along with the text features $\mathbf{T}$, into a common alignment space d_a, yielding $\mathbf{V}_0$ and $\mathbf{T}_0$. Subsequently, a two-layer bidirectional cross-attention is performed in two directions. (1) Vision $\rightarrow$ Text: Using $\mathbf{V}_0$ as Query and $\mathbf{T}_0$ as Key/Value, allowing each visual position to attend to the most relevant semantic words, which can be formulated as

$$\mathbf{V}' = \text{CrossAttention}(\mathbf{Q} = \mathbf{V}_0, \mathbf{K} = \mathbf{T}_0, \mathbf{V} = \mathbf{T}_0). \tag{7}$$

(2) Text $\rightarrow$ Vision: Using $\mathbf{T}_0$ as Query and the updated $\mathbf{V}'$ as Key/Value, allowing each word to focus on the most relevant visual region, which can be expressed as

$$\mathbf{T}' = \text{CrossAttention}(\mathbf{Q} = \mathbf{T}_0, \mathbf{K} = \mathbf{V}', \mathbf{V} = \mathbf{V}'). \tag{8}$$

The aligned visual features $\mathbf{V}'$ are reshaped and projected into LLaVA embedding space, then concatenated with the aligned text features $\mathbf{T}'$ to form the input sequence $\mathbf{X}_{\text{llava}}$ for LLaVA, laying a solid foundation for generating accurate and detailed descriptions.

3.4. Efficient Parameter Fine-tuning Strategy

To avoid the high computational cost and overfitting risk associated with full-parameter fine-tuning, we employ LoRA for efficient fine-tuning of LLaVA. Specifically, LoRA adapters are injected only into the linear layers of the self-attention (Q, V projections) and Feed-Forward Network (FFN) within each Transformer layer of LLaVA. The rank is set to $r = 16$ and the scaling factor to $\alpha = 32$. The visual encoder and visual projection layers of LLaVA remain frozen. This strategy enables rapid adaptation of the model to the defect diagnosis task with a minimal number of parameters.

3.5. Loss Functions

SAM-LLaVA is optimized end-to-end via a multi-task loss. The total loss function is defined as a weighted sum of the segmentation loss and the text generation loss:

$$\mathcal{L}_{\text{total}} = \alpha \mathcal{L}_{\text{seg}} + \gamma \mathcal{L}_{\text{gen}}, \tag{9}$$

where the balancing coefficients α and γ are both set to 1.0. $\mathcal{L}_{\text{seg}}$ is the segmentation loss for the multi-scale mask decoder as defined in Eq. (6), optimizing defect localization accuracy. $\mathcal{L}_{\text{gen}}$ is the standard cross-entropy loss for auto-regressive language modeling, supervising the generation of defect descriptions:

$$\mathcal{L}_{\text{gen}} = -\sum_{i=1}^{L_{\text{ans}}} \log P(\hat{t}_i | \hat{t}_{<i}, \mathbf{X}_{\text{llava}}), \tag{10}$$

where $\hat{\mathbf{T}} = \{\hat{t}_i\}$ is the sequence of tokens generated by the model, and $\mathbf{X}_{\text{llava}}$ is the aligned input sequence to LLaVA.

During the training stage, we freeze the image encoders of CLIP and SAM, optimizing only SAM prompt encoder and mask decoder (including the new multi-scale parts), the segmentation-aware semantic alignment module, the projection layers, and the LoRA parameters. By jointly optimizing $\mathcal{L}_{\text{total}}$, SAM-LLaVA achieves synergistic improvement in the performance of both defect detection and description generation.

4. Experimental Setup

Here, the baselines, data sets, performance metrics and implementation notes utilized in our experiments are introduced in turn.

4.1. Baselines

In total, 13 state-of-the-art approaches, including AA-CLIP [33], AdaCLIP [34], AnomalyCLIP [35], AnomalyGPT [14], Myriad [15], PaDiM [36], PatchCore [5], PromptAD [37], SPADE [6], ADPretrain [38], SSVP [39], SPGDD-GPT [12] and WinCLIP [23], were used in the quantitative evaluation experiment. These approaches can be divided into three classes, i.e., conventional deep defect detection approaches, LVLM-based defect detection approaches and image-text contrastive learning-based

defect detection approaches. For fair comparison, we directly obtained the results of the baselines reported in SPGDD-GPT [12], because these baselines were trained and evaluated on the same TADD data set under identical protocols. This choice ensures a consistent and fair evaluation across different methods.

Given the approaches which originally utilized few-shot settings, we employed the original configurations. In the case where the approaches did not apply few-shot learning, for example, AdaCLIP [34] and AnomalyCLIP [14], we utilized the settings reported in [14]. Considering that fine-tuning need to be used for the LVLM-based approaches such as AnomalyGPT [14], Myriad [15] and SPGDD-GPT [12], the original fine-tuning schemes and settings were used. The Adam optimizer was used to train those baselines with a learning rate of $1e-4$. The batch size was set to 8 and each model was trained for 50 epochs.

4.2. Data Sets

Following previous research [14, 12], we performed the zero-shot defect detection experiment using the MVTec-AD [19] and VisA [20] data sets. Totally, 1,725 abnormal images and 3,629 normal images were included in MVTec-AD [19], which were categorized into 15 classes. On the other hand, 1,200 abnormal images and 8,821 normal images were comprised of the VisA [20] data set, covering 12 classes. The partitioning scheme and random seed ($SEED = 42$) utilized in [12] were employed for both data sets.

We conducted the few-shot defect detection experiment using the TADD [12]. This data set consists of around 19,500 abnormal images and 16,000 normal images, spanning over 30 classes. The TADD was divided into the training, validation and testing sets at the percentages of 70%, 15% and 15%, respectively. We applied the dividing to each class of defects of the 21 subsets. In this case, the testing set contained all classes of defects. If a class had no more than 50 images, however, at least 10 images of this class were included in the testing set. This choice guaranteed the reliability of the evaluation experiment.

4.3. Performance Metrics

To assess the performance of defect detection methods, we utilized the Area Under the Receiver Operating Characteristic (AUROC) or the Area Under the Curve (AUC) metric. This metric was known as Image-AUC and Pixel-AUC for the image-level and pixel-level tasks, respectively. For LVLM-based approaches, we additionally measure image-level accuracy, defined as whether the LLM's textual response correctly classifies the entire image as containing a defect or being normal (i.e., a binary decision).

4.4. Implementation Notes

For the LLMs used in both inference and training stages, the CLIP model was adopted from the ViT-B/16 [4] version pre-trained on the LAION-400M [40] data set, in which the dimension of feature embeddings outputted by the image encoder was 512. The SAM model used the officially released ViT-H [4] backbone, where the image encoder produced feature maps with spatial dimensions of 64×64 and a channel dimension of 768. The language model in LLaVA utilized Vicuna-7B-v1.5 [31], and image features were mapped into the 4096-D embedding space of the language model through a linear projection layer. Regarding the Vicuna-7B-v1.5 model, parameter-efficient fine-tuning was performed using LoRA, specifically by injecting LoRA adapters into the query (Q) and value (V) projection matrices of the self-attention module as well as the two fully-connected layers of the Feed-Forward Network (FFN) in each Transformer layer, with rank $r = 16$ and scaling factor $\alpha = 32$. Matrix $\mathbf{A}$ was initialized using Kaiming normal distribution [41], while matrix $\mathbf{B}$ was initialized as a zero matrix, ensuring that the model behavior remained consistent with the pre-trained model at the initial training stage.

During the training stage of SAM-LLaVA, we used the Adam optimizer with an initial learning rate of 1×10^{-4}. The batch size was set to 8, and the model was trained for 50 epochs. The entire model was end-to-end trained on two NVIDIA L40 GPUs. These settings were kept consistent across all zero-shot and few-shot experiments unless otherwise specified.

5. Experimental Results

We conducted a quantitative evaluation experiment, a qualitative analysis and an ablation study in turn. In this section, we report the results derived in these experiments.

5.1. Quantitative Evaluation

Both the zero-shot and the few-shot defect detection experiments were performed. The results obtained are presented separately in this subsection.

5.1.1. Zero-Shot Defect Detection

The proposed approach was compared against 13 baselines for the zero-shot defect detection task. Using the MVTec-AD [19] and the VisA [20] data sets, two experiments were carried out, respectively. Following the previous study [12], we trained a model using the VisA [20] data set and applied this model to the MVTec-AD [19] data set, and vice versa. Table 1 reports the results derived. As can be observed, our SAM-LLaVA always achieved the best performance in the zero-shot task. On VisA $\rightarrow$ MVTec-AD, the Image-AUC and Pixel-AUC values derived using our method reached 94.8% and 95.6%, respectively, surpassing the strongest baseline, i.e., SPGDD-GPT, (93.6% Image-AUC, 94.8% Pixel-AUC) by 1.2% and 0.8%. In contrast to the state-of-the-art SPGDD-GPT, the performance gain should stem primarily from the pixel-precise segmentation capability of SAM, which overcomes the patch-granularity limitation inherent in the TDDF of SPGDD-GPT. On MVTec-AD $\rightarrow$ VisA, our SAM-LLaVA also outperformed all 13 baseline methods.

5.1.2. Few-Shot Defect Detection

We trained the proposed SAM-LLaVA and the 13 baselines using the TADD [12] in the few-shot experiment. Each model was tested using a single subset of the TADD. The Pixel-AUC and accuracy metrics were used to assess the performances of defect detection and defect description, respectively. Three few-shot settings were utilized in this experiment, including 1-shot, 2-shot and 4-shot. Tables 2, 3 and 4 show the Pixel-AUC values that the 13 baselines and our method derived in the three settings,

Table 1: Comparison between 13 baselines and the proposed SAM-LLaVA in Image-AUC and Pixel-AUC for the zero-shot defect detection task. In particular, the data sets positioned at the left and right sides of the arrow "→" stand for the training and testing data sets, respectively. Note that the results of baselines are directly derived from [12]. (Since Myriad reused the image encoder, anomaly scoring scheme and pre-trained weights of AnomalyGPT for anomaly map generation, the identical Image-AUC and Pixel-AUC scores were derived using them). The best result is indicated in the bold fonts with regard to each metric. This continues for the following tables.

Method	VisA [20] → MVTec-AD [19]		MVTec-AD [19] → VisA [20]	
	Image-AUC	Pixel-AUC	Image-AUC	Pixel-AUC
SPADE [6]	77.9	80.4	75.3	77.8
PaDiM [36]	73.8	75.2	59.1	60.4
PatchCore [5]	79.5	82.8	73.8	76.7
WinCLIP [23]	91.8	85.1	76.4	78.1
PromptAD [37]	81.8	83.2	74.6	87.5
AnomalyGPT [14]	93.2	94.6	85.1	93.8
Myriad [15]	93.2	94.6	85.1	93.8
AnomalyCLIP [35]	91.5	91.1	82.1	95.4
AA-CLIP [33]	90.5	91.9	84.6	95.5
AdaCLIP [34]	90.0	89.9	84.3	95.5
SPGDD-GPT [12]	93.6	94.8	85.5	94.2
ADPretrain [38]	93.4	94.7	86.7	95.5
SSVP [39]	93.0	92.2	86.7	95.6
SAM-LLaVA (Ours)	**94.8**	**95.6**	**86.9**	**95.7**

respectively. It can be seen that our SAM-LLaVA consistently produced the best or competitive results on most of the 21 subsets across all three settings.

To better illustrate the overall trends, we further computed the average Pixel-AUC value across all 21 subsets for each method. As shown in Fig. 2, SAM-LLaVA achieved average scores of 93.3%, 94.0%, and 95.0% under the 1-shot, 2-shot, and 4-shot settings, respectively. These results consistently outperformed SPGDD-GPT [12] by margins of 2.0%, 1.4%, and 1.3%.

Following the same protocol used for SPGDD-GPT [12], we also evaluated the

Table 2: Comparison between the 13 baselines and our SAM-LLaVA in Pixel-AUC obtained using the 21 subsets of the TADD [12] for the 1-shot defect detection task. We performed the experiment for five runs and calculated the average Pixel-AUC value across these runs. Note that the results of baselines are directly derived from [12]. (Since Myriad reused the image encoder, anomaly scoring scheme and pre-trained weights of AnomalyGPT for anomaly map generation, the identical Image-AUC and Pixel-AUC scores were derived using them).

Subset	Method													
	SPADE [6]	PaDiM [36]	PatchCore [5]	WinCLIP [23]	PromptAD [37]	AnomalyGPT [14]	Myriad [15]	AnomalyCLIP [35]	AA-CLIP [33]	AdaCLIP [34]	SPGDD-GPT [12]	ADPretrain [38]	SSVP [39]	SAM-LLaVA (Ours)
Aitex	48.6	68.8	63.6	92.8	69.1	85.2	85.2	82.1	78.8	73.4	93.2	88.6	75.4	**95.6**
BSData	56.8	63.7	50.6	75.6	74.8	80.3	80.3	79.3	76.8	80.6	85.6	87.4	83.2	**90.2**
RubberTires	66.3	70.5	59.7	80.4	71.6	89.7	89.7	87.2	86.9	76.3	93.4	93.7	83.3	**96.3**
CFD	64.9	74.0	87.5	90.0	93.2	94.0	94.0	93.1	92.8	89.3	94.8	92.9	89.0	**95.4**
Crack500	55.0	71.3	65.8	74.5	83.8	86.8	86.8	85.5	85.1	84.2	87.1	82.6	85.7	**91.1**
CrackTree200	48.8	64.6	80.2	86.4	89.7	90.6	90.6	89.2	88.9	88.5	91.5	84.1	88.1	**92.7**
DeepCrack	62.6	82.1	66.5	82.8	86.7	89.9	89.9	80.4	88.0	79.0	**90.5**	89.7	76.9	89.6
Eugen_Miller	73.7	80.2	82.5	87.2	92.8	92.6	92.6	91.7	91.3	89.1	94.5	91.6	89.1	**95.5**
GAPs	50.7	62.9	55.3	75.0	85.6	96.4	96.4	96.0	95.5	96.4	**97.5**	95.1	96.3	97.4
KolektorSDD	54.7	79.5	74.8	89.7	90.8	71.3	71.3	87.2	69.8	88.6	91.0	86.3	87.6	**93.2**
KolektorSDD2	61.8	71.8	85.8	87.6	87.7	70.4	70.4	86.5	69.1	87.2	88.6	86.0	87.2	**93.0**
LIACI	51.2	61.9	50.8	77.3	63.9	82.5	82.5	62.4	80.9	81.7	86.7	80.3	85.6	**88.4**
MT	63.2	85.7	83.2	92.8	92.4	84.6	84.6	83.2	82.8	84.0	88.8	83.9	90.7	**93.6**
MVTec-AD	91.2	89.3	92.0	95.2	85.9	95.3	95.3	90.6	89.9	87.6	95.8	95.9	92.8	**96.1**
NEU	57.3	89.3	62.3	91.5	60.9	95.6	95.6	78.8	94.8	83.2	95.6	74.6	85.5	**95.8**
OUCCrack	54.9	68.8	57.9	55.5	57.3	90.0	90.0	65.5	76.4	60.1	91.8	87.8	91.7	**93.2**
Rissbilder	66.3	70.5	59.7	80.4	71.6	89.7	89.7	87.2	76.8	74.3	93.4	78.2	83.1	**94.6**
RSDD	87.5	81.7	60.8	84.4	71.4	72.2	72.2	71.0	70.6	71.5	85.8	71.5	76.4	**89.4**
RSDD2	83.4	78.0	54.7	70.8	61.1	70.3	70.3	69.4	69.0	69.8	83.9	71.6	76.8	**89.9**
Visa	95.6	89.9	95.4	96.4	89.0	96.2	96.2	89.8	91.9	81.2	96.8	**96.9**	96.7	96.5
Volker	59.2	65.4	54.1	83.2	84.9	90.5	90.5	89.8	79.5	85.6	91.0	85.4	84.1	**92.7**

Table 3: Comparison between the 13 baselines and our SAM-LLaVA in Pixel-AUC obtained using the 21 subsets of the TADD [12] for the 2-shot defect detection task. We performed the experiment for five runs and calculated the average Pixel-AUC value across these runs. Note that the results of baselines are directly derived from [12]. (Since Myriad reused the image encoder, anomaly scoring scheme and pre-trained weights of AnomalyGPT for anomaly map generation, the identical Image-AUC and Pixel-AUC scores were derived using them).

Subset	Method													
	SPADE [6]	PaDiM [36]	PatchCore [5]	WinCLIP [23]	PromptAD [37]	AnomalyGPT [14]	Myriad [15]	AnomalyCLIP [35]	AA-CLIP [33]	AdaCLIP [34]	SPGDD-GPT [12]	ADPretrain [38]	SSVP [39]	SAM-LLaVA (Ours)
Aitex	59.8	69.3	65.2	92.9	71.4	87.2	87.2	83.4	80.1	73.6	93.6	89.0	75.6	**95.9**
BSData	52.6	68.4	58.7	68.8	72.6	84.6	84.6	81.2	78.4	80.1	88.5	87.5	83.8	**90.8**
RubberTires	55.8	73.2	65.9	74.8	81.4	91.5	91.5	88.9	88.3	77.8	93.7	93.8	83.6	**96.7**
CFD	66.0	74.5	87.5	91.3	94.8	95.8	95.8	94.5	94.0	90.5	**96.3**	93.0	90.3	95.8
Crack500	55.9	72.9	67.0	74.8	84.0	88.1	88.1	87.1	86.6	85.4	89.5	83.1	85.9	**91.7**
CrackTree200	50.0	64.5	81.4	87.0	91.9	91.9	91.9	90.7	90.2	89.6	92.7	85.2	88.7	**93.5**
DeepCrack	63.0	82.9	67.0	83.5	88.5	91.2	91.2	81.0	89.5	80.5	**91.8**	90.2	78.3	90.0
Eugen_Miller	75.2	82.2	82.4	88.6	93.0	93.9	93.9	92.9	92.4	90.3	95.0	91.7	89.5	**96.0**
GAPs	52.3	63.1	58.2	76.6	85.8	97.5	97.5	97.2	96.8	97.4	**97.8**	95.3	96.7	**97.8**
KolektorSDD	56.4	82.1	76.3	90.6	90.8	72.6	72.6	87.6	71.3	89.3	92.4	86.4	87.7	**93.8**
KolektorSDD2	62.0	73.2	86.9	89.1	88.4	72.4	72.4	87.2	70.7	88.4	90.3	86.9	87.8	**93.2**
LIACI	58.7	64.6	60.4	70.5	75.6	85.3	85.3	65.1	82.6	83.7	**89.4**	80.5	85.7	88.8
MT	63.1	86.0	83.7	93.8	93.2	86.7	86.7	84.7	84.2	85.6	90.8	84.2	91.0	**94.1**
MVTec-AD	92.0	91.3	93.3	96.0	86.4	95.6	95.6	90.8	91.0	88.9	95.9	96.5	93.4	**96.7**
NEU	60.1	89.8	63.0	93.2	62.7	96.0	96.0	80.2	95.0	84.6	96.7	74.9	85.8	**96.8**
OUCCrack	56.7	80.2	58.4	65.9	57.9	92.3	92.3	66.1	76.8	62.6	93.2	88.1	92.0	**94.2**
Rissbilder	55.8	73.2	57.2	74.8	81.4	91.5	91.5	88.5	78.0	74.5	93.7	78.5	83.3	**95.1**
RSDD	89.1	81.8	62.4	85.2	73.6	74.8	74.8	72.5	72.0	73.6	87.0	71.6	76.6	**90.6**
RSDD2	83.8	77.9	55.2	72.2	63.0	72.0	72.0	70.6	70.1	71.4	85.6	71.8	77.1	**91.2**
Visa	96.2	92.0	96.1	96.8	89.8	96.4	96.4	90.2	93.4	80.9	97.2	**97.3**	96.9	96.8
Volker	62.3	65.4	55.8	84.3	85.7	91.6	91.6	90.4	80.9	85.9	93.8	85.8	84.2	**93.9**

Table 4: Comparison between the 13 baselines and our SAM-LLaVA in Pixel-AUC obtained using the 21 subsets of the TADD [12] for the 4-shot defect detection task. We performed the experiment for five runs and calculated the average Pixel-AUC value across these runs. Note that the results of baselines are directly derived from [12]. (Since Myriad reused the image encoder, anomaly scoring scheme and pre-trained weights of AnomalyGPT for anomaly map generation, the identical Image-AUC and Pixel-AUC scores were derived using them).

Subset	Method													
	SPADE [6]	PaDiM [36]	PatchCore [5]	WinCLIP [23]	PromptAD [37]	AnomalyGPT [14]	Myriad [15]	AnomalyCLIP [35]	AA-CLIP [33]	AdaCLIP [34]	SPGDD-GPT [12]	ADPretrain [38]	SSVP [39]	SAM-LLaVA (Ours)
Aitex	60.6	71.7	66.8	93.4	73.7	88.4	88.4	85.3	80.6	76.2	94.9	90.3	75.9	**96.5**
BSData	54.2	69.5	59.8	69.6	74.5	85.4	85.4	82.4	81.7	83.1	89.8	87.6	84.2	**91.9**
RubberTires	57.4	74.8	67.5	75.4	83.1	92.8	92.8	90.1	89.6	79.9	94.5	94.1	83.7	**97.1**
CFD	66.4	75.6	88.3	93.0	96.0	96.5	96.5	95.0	95.3	92.3	**97.4**	93.1	90.5	96.7
Crack500	57.3	74.3	68.3	76.2	85.5	89.2	89.2	88.9	88.3	86.9	90.3	83.2	86.3	**92.6**
CrackTree200	51.7	64.8	81.8	87.8	92.1	93.0	93.0	92.2	92.0	91.4	93.7	85.5	88.9	**94.5**
DeepCrack	64.1	84.2	67.9	83.8	90.3	92.6	92.6	82.0	91.1	82.8	**93.2**	90.6	78.6	91.3
Eugen_Miller	75.8	84.7	84.3	89.7	93.5	94.1	94.1	93.8	92.6	90.8	95.4	91.8	89.7	**97.3**
GAPs	53.2	64.4	60.6	77.3	86.4	98.0	98.0	97.6	97.0	98.0	98.4	95.4	96.8	**98.5**
KolektorSDD	57.9	84.0	76.8	91.6	91.5	74.3	74.3	89.5	73.0	90.3	92.8	86.7	87.9	**95.1**
KolektorSDD2	63.2	74.6	87.4	90.7	89.7	74.5	74.5	88.4	72.5	89.1	92.6	87.1	88.0	**94.6**
LIACI	60.1	66.3	61.6	71.4	76.8	86.9	86.9	70.4	84.7	86.0	**90.5**	80.8	85.8	89.7
MT	65.0	87.0	84.9	95.7	94.8	87.2	87.2	87.1	86.6	87.9	92.6	84.3	91.5	**95.4**
MVTec-AD	92.7	92.6	94.3	96.2	87.4	96.2	96.2	92.4	91.1	89.8	96.8	96.7	93.5	**97.5**
NEU	62.0	90.3	64.5	94.5	63.6	96.8	96.8	81.6	95.2	86.5	**97.3**	75.2	86.0	97.1
OUCCrack	58.4	82.5	60.1	57.9	59.5	93.4	93.4	69.4	78.5	65.9	94.7	88.6	92.4	**95.8**
Rissbilder	57.4	74.8	67.5	75.4	83.1	92.8	92.8	90.2	80.1	75.6	94.5	78.9	83.7	**96.3**
RSDD	91.8	83.0	63.8	86.5	74.6	76.4	76.4	75.1	74.6	76.2	89.4	71.8	76.7	**91.9**
RSDD2	84.5	79.0	56.5	73.4	64.5	73.6	73.6	73.0	72.5	74.0	87.2	72.2	77.5	**91.8**
Visa	96.6	93.2	96.8	97.2	90.3	96.7	96.7	91.3	93.6	83.5	97.3	**97.4**	97.0	**97.4**
Volker	63.9	66.1	57.4	85.6	87.0	92.4	92.4	90.8	81.2	86.6	94.1	85.9	84.2	**95.0**

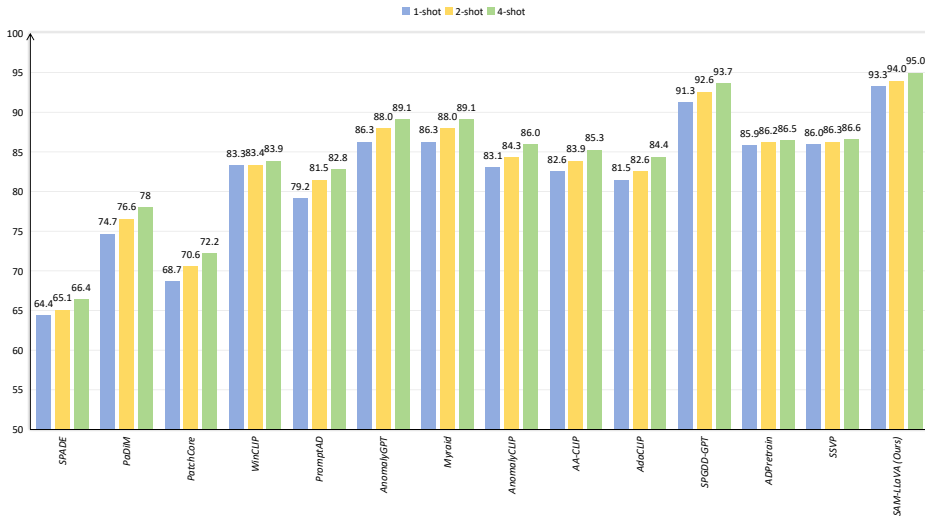

Figure 2: In terms of the 1-shot, 2-shot and 4-shot settings, the average Pixel-AUC values obtained using one of the 13 baselines and our approach across the TADD [12] are shown in each group.

accuracy of defect description derived using three baselines and our method. For the fair comparison purpose, only the MVTec-AD [19] and VisA [20] data sets were used for testing. Table 5 reports the results produced by the four methods at three settings. Using the 1-shot setting, our SAM-LLaVA achieved accuracy values of 92.3% and 83.8% on MVTec-AD and VisA, respectively, surpassing SPGDD-GPT by 2.2% and 3.2%. At the 2-shot and 4-shot settings, SAM-LLaVA continued to outperform all the baselines, demonstrating its ability to generate more accurate and detailed textual descriptions of defects even with very few training samples.

It should be noted that the image-level description accuracy on MVTec-AD exhibits a slight decrease from 1-shot (92.3%) to 4-shot (90.8%), as shown in Table 5. This finding reflects a known trade-off in LVLM-based few-shot industrial anomaly diagnosis. That is to say, increasing the number of normal shots improves the accuracy of defect localization (Pixel-AUC increases from 96.1% to 97.5%), while it may inadvertently broaden the learned semantic boundary of normality of the LLM due to richer input features. Consequently, the model becomes slightly more conservative, leading to marginal false negatives on subtle or near-boundary defects. This trend is consistent

Table 5: Comparison of the accuracy values obtained using three baselines and our SAM-LLaVA on the MVTec-AD [19] and VisA VisA [20] data sets with different few-shot setups. Note that the results of baselines are directly derived from [12].

Setting	Method	MVTec-AD	VisA
1-Shot	AnomalyGPT [14]	86.1 ± 1.1	77.4 ± 1.0
	Myriad [15]	87.4 ± 0.9	80.5 ± 1.2
	SPGDD-GPT [12]	90.1 ± 1.9	80.6 ± 1.7
	SAM-LLaVA (Ours)	**92.3 ± 1.2**	**83.8 ± 1.0**
2-Shot	AnomalyGPT [14]	84.8 ± 0.8	77.5 ± 0.3
	Myriad [15]	85.4 ± 0.7	78.9 ± 0.5
	SPGDD-GPT [12]	89.0 ± 2.3	79.5 ± 1.6
	SAM-LLaVA (Ours)	**92.0 ± 1.6**	**82.9 ± 1.5**
4-Shot	AnomalyGPT [14]	85.0 ± 0.3	77.7 ± 0.4
	Myriad [15]	85.3 ± 0.3	78.3 ± 0.3
	SPGDD-GPT [12]	88.7 ± 2.3	78.9 ± 1.7
	SAM-LLaVA (Ours)	**90.8 ± 2.1**	**81.1 ± 1.9**

with observations in AnomalyGPT [14] and Myriad [15], which show similar behaviors to internal variations among normal samples.

5.2. Qualitative Analysis

To evaluate the quality of the textual descriptions generated using five approaches, including PandaGPT [30], MiniGPT-4 [42], AnomalyGPT [14], SPGDD-GPT [12] and the proposed SAM-LLaVA, we applied an LLM-as-a-Judge scheme to each image contained in the TADD [12] at the zero-shot setting. To be specific, each image-text pair was sent to the pre-trained GPT-4V [43] model. This model was prompted to generate a score in the range of [0,100] according to four criteria: correctness, completeness, fine-grained details (e.g., position, color, shape, category), and interpretability. Given an image which did not contain a defect, we also assessed the sufficiency of the description of normal content.

Table 6: Comparison between four baselines and our SAM-LLaVA in GPT-4V [43] score, where "Abnormal", "Normal" and "All" represent the average scores of the abnormal, normal and both abnormal and normal graphs.

Method	Abnormal	Normal	All
PandaGPT [30]	30.0	40.0	35.0
MiniGPT-4 [42]	21.5	61.5	41.5
AnomalyGPT [14]	80.5	86.5	83.5
SPGDD-GPT [12]	94.6	93.4	94.0
SAM-LLaVA (Ours)	**95.5**	**96.5**	**96.0**

We chose GPT-4V as the evaluator for the following reasons. First, unlike text-only LLMs, GPT-4V is a multimodal model that can simultaneously process both the image and the textual description, allowing it to assess image-text consistency (e.g., whether the described defect location, color, and shape actually exist in the image). Second, recent studies [44, 45] have demonstrated that GPT-4V is able to achieve high correlation with human judgments on visual description evaluation tasks, making it a reliable proxy for human evaluation. Third, GPT-4V supports flexible scoring criteria, allowing us to define four task-relevant dimensions, including correctness (whether the description accurately reflects the defect), completeness (whether all defect attributes are covered), fine-grained details (e.g., position, color, shape and category), and interpretability (whether the description is clear and actionable for human inspectors).

Table 6 reports the average scores computed across all descriptions produced by each approach. It can be seen that PandaGPT, MiniGPT-4, AnomalyGPT, SPGDD-GPT and SAM-LLaVA produced average scores of 35.0, 41.5, 83.5, 94.0 and 96.0, respectively. The highest score was achieved by our SAM-LLaVA, which suggests its superior ability to generate accurate, complete, and detailed textual descriptions, even compared to the latest baseline SPGDD-GPT [12].

To further illustrate the performance gap, we present two representative examples. The first example involves an image of abnormal fabric in Fig. 3. PandaGPT [30] incorrectly hallucinated an unrelated "finger" in the image, entirely misunderstanding the scene. MiniGPT-4 [42] failed to detect any anomaly, producing a false negative.

AnomalyGPT [14] correctly localized the defect to the bottom-left region and identified the object as fabric, but provided no further descriptive details such as color, shape, or defect type. SPGDD-GPT [12] significantly improved upon this by correctly describing the fabric as "gray braided" and the defect as a "black scratch" in the center, resembling a knife cut. As an end-to-end feature alignment method, however, SPGDD-GPT lacks access to pixel-level segmentation boundaries and thus cannot describe fine-grained spatial attributes. SAM-LLaVA further enhanced the description by specifying the scratch as "narrow, linear", noting its precise central location, describing its "sharp edges" and "depth contrast" which are details that require explicit pixel-mask grounding through the SASA module, and formally concluding that no other defects are present. This demonstrates the superior ability of SAM-LLaVA to produce more precise, complete, and professionally structured defect descriptions, a direct benefit of its segmentation-aware semantic alignment mechanism.

The second example shows a normal pill image in Fig. 4. PandaGPT again hallucinated severely, describing an implausible "cat on a plate" scenario. MiniGPT-4 correctly identified no defect but inaccurately described the pill's marking as the letter "f" in red ink. AnomalyGPT provided a correct but minimal response, stating only "This is a white pill." SPGDD-GPT offered a much more detailed and accurate description, noting the black background, the pill's central placement, the "FF" logo, and the presence of red dots. SAM-LLaVA built upon this foundation with even greater precision: it described the pill's shape ("oval-shaped"), surface quality ("smooth and uniform"), the logo's font ("serif font"), and the nature of the red dots ("circular ... part of the normal manufacturing markings"). It also explicitly affirmed the pill's integrity, listing the absence of "cracks, discolorations, or deformities." This comparison highlights SAM-LLaVA's enhanced capability for accurate, exhaustive, and non-hallucinatory description of normal industrial samples.

To further demonstrate the superiority of our SAM-LLaVA in defect segmentation, we present a visual comparison of segmentation masks and intermediate coarse heatmaps in Fig. 5. The first column shows the input image, the second column presents the ground-truth (GT) mask, followed by segmentation results from WinCLIP [23], AnomalyGPT [14], SPGDD-GPT [12], and our SAM-LLaVA. The last column dis-

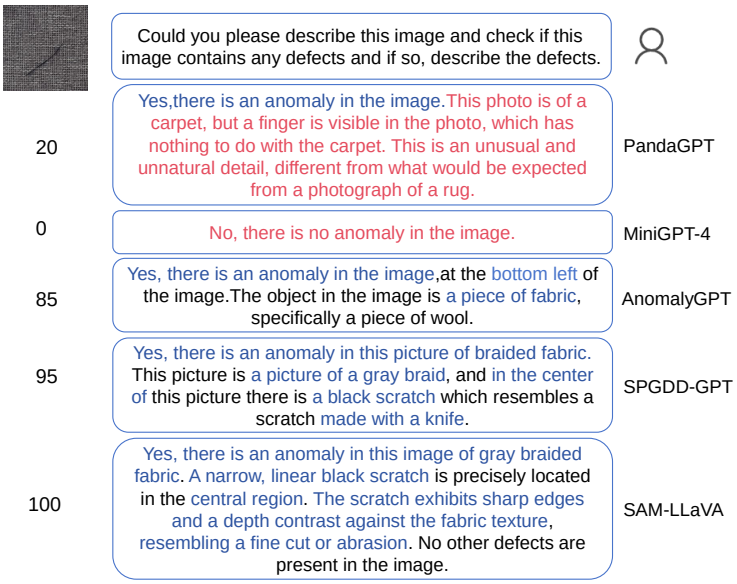

Figure 3: Comparison between four baselines, including PandaGPT [30], MiniGPT-4 [42], AnomalyGPT [14] and SPGDD-GPT [12], and our SAM-LLaVA in textual description that they generated in terms of a fabric image with a scratch. Regarding each textual description, a score generated by GPT-4V [43] is shown at its left side. (Note that the results of baselines are directly derived from [12].) Although SPGDD-GPT already provided a detailed description of the defect, our SAM-LLaVA further enhanced the granularity by precisely describing the scratch's geometry (e.g., "narrow, linear"), edge characteristics, and confirming the absence of other defects, demonstrating its superior segmentation-aware semantic alignment.

plays the coarse heatmap generated by our CLIP-based coarse localization module. Compared to existing methods, SAM-LLaVA produced segmentation masks that are more precise, continuous, and closely aligned with the GT boundaries, especially for fine-grained and irregular defects. The coarse heatmap also demonstrates the effectiveness of our CLIP-SAM cascade in providing semantically meaningful guidance for subsequent fine segmentation.

These qualitative results confirm that SAM-LLaVA not only excels in defect detection and segmentation but also sets a new standard for generating faithful, informative, and human-readable descriptions. By effectively bridging the gap between high-precision visual perception and detailed linguistic explanation, SAM-LLaVA delivers a more reliable and transparent diagnostic output suitable for industrial inspection sce-

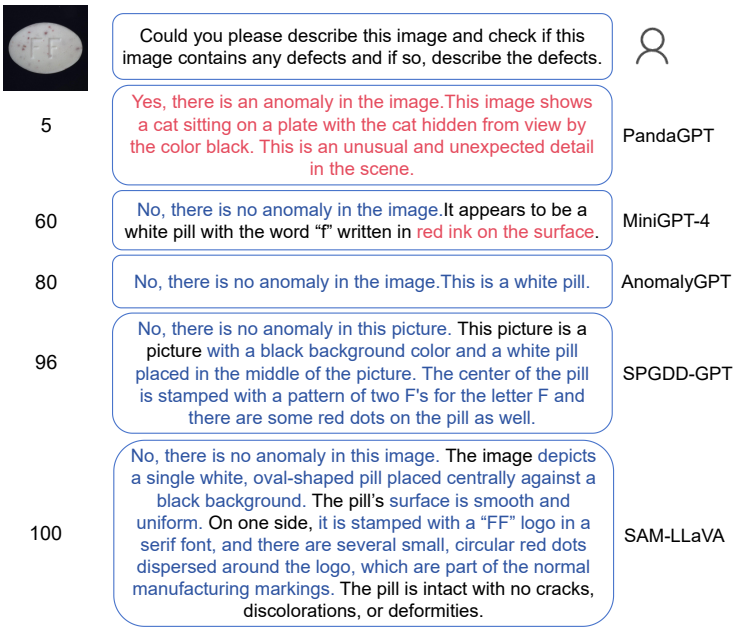

Figure 4: Comparison between four baselines, including PandaGPT [30], MiniGPT-4 [42], AnomalyGPT [14] and SPGDD-GPT [12], and our SAM-LLaVA in textual description generated in terms of a normal image containing a pill. For each description, a score generated by GPT-4V [43] is shown at its left side. (Note that the results of baselines are directly derived from [12].) Compared to all baselines, SAM-LLaVA provided the most accurate, structured, and hallucination-free description, detailing the pill's shape, surface texture, imprint features, and explicitly stating its integrity without anomalies.

narios.

5.3. Ablation Study

To investigate the effectiveness of the proposed components in SAM-LLaVA, we conducted a comprehensive series of ablation experiments. Specifically, we examined the impact of the CLIP-SAM cascade prompting, the multi-scale prompt learner and mask decoder, and the Segmentation-Aware Semantic Alignment (SASA) module. Additionally, we compared different prompt encoding strategies. For clarity, all ablation studies were conducted by training the variants on the TADD [12] data set and evaluating them on the MVTec-AD [19] and VisA [20] data sets under the 1-shot setting unless otherwise specified.

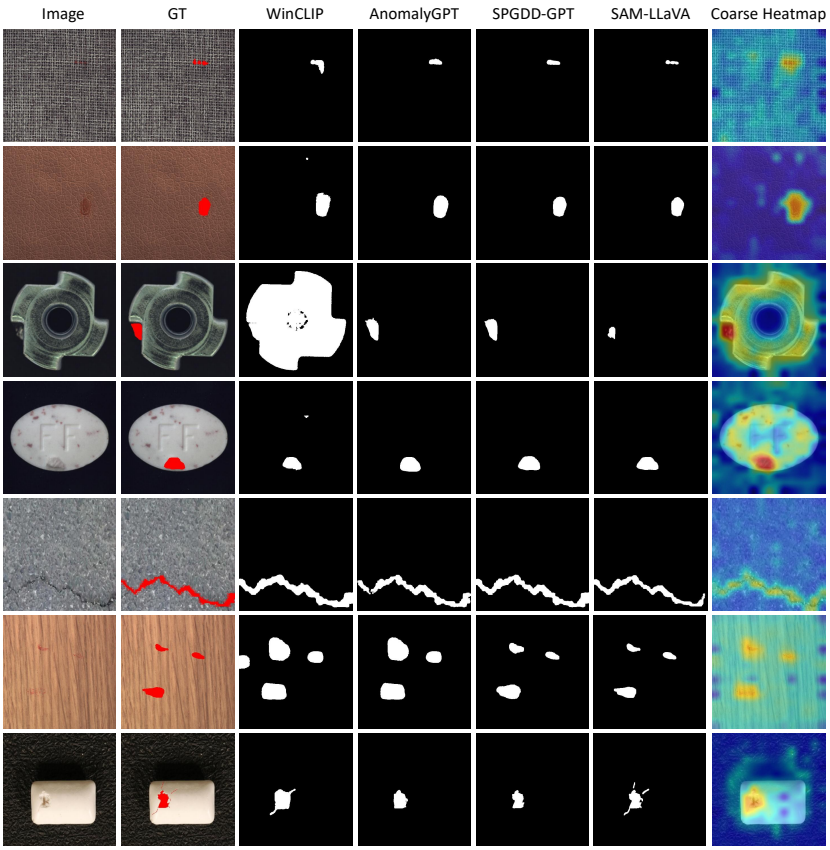

Figure 5: Visual comparison of segmentation results across different methods. From left to right: original image, ground-truth mask, segmentation results from WinCLIP [23], AnomalyGPT [14], SPGDD-GPT [12], SAM-LLaVA, and the coarse heatmap from SAM-LLaVA. Our method consistently produces more accurate and detailed segmentation masks, especially for fine-grained and irregular defects. The coarse heatmap illustrates the initial localization capability of the CLIP-SAM cascade.

5.3.1. Impact of the CLIP-SAM Cascade Prompting

We first evaluated the necessity of the coarse-to-fine cascade design. A variant (w/o Cascade) was created by removing the CLIP-based coarse prompt generation and directly using SAM with its original image encoder without any geometric prompts. As shown in Table 7, the removal of the cascade mechanism led to a significant drop in both Pixel-AUC and Image-AUC across both data sets, with decreases of 3.0% and 4.5% in Pixel-AUC on MVTec-AD and VisA, respectively. This finding confirms that

Table 7: Comparison between SAM-LLaVA and its variant without CLIP-SAM Cascade in Image-AUC, Pixel-AUC and accuracy (Acc.) derived on the MVTec-AD [19] and VisA [20] data sets.

Model Variant	MVTec-AD			VisA		
	Image-AUC	Pixel-AUC	Acc.	Image-AUC	Pixel-AUC	Acc.
SAM-LLaVA	**95.3**	**96.1**	**92.3**	**88.2**	**96.5**	**83.8**
w/o CLIP-SAM Cascade	92.5	93.1	89.5	85.0	92.0	79.8

Table 8: Comparison between SAM-LLaVA with and its three variants without multi-scale components in Image-AUC, Pixel-AUC and accuracy (Acc.) derived on the MVTec-AD [19] and VisA [20] data sets.

Model Variant	MVTec-AD			VisA		
	Image-AUC	Pixel-AUC	Acc.	Image-AUC	Pixel-AUC	Acc.
SAM-LLaVA	**95.3**	**96.1**	**92.3**	**88.2**	**96.5**	**83.8**
w/o MS-Prompt	93.7	93.5	90.8	86.2	95.3	82.0
w/o MS-Decoder	94.3	94.7	91.2	86.5	94.8	82.3
Single-scale SAM	92.8	93.0	89.4	82.8	90.9	78.6

the CLIP-generated coarse prompts are crucial for guiding SAM to focus on semantically suspicious regions, especially for subtle and low-contrast defects that SAM alone might overlook.

5.3.2. Impact of the Multi-scale Design

To assess the contribution of the multi-scale components, we designed three variants by removing the multi-scale prompt learner (w/o MS-Prompt), removing the multi-scale mask decoder (w/o MS-Decoder), and replacing multi-scale prompt learner and mask decoder by a single-scale SAM. The first two variants use native prompt learners and mask decoders instead. The results are presented in Table 8. Removing the multi-scale prompt learner reduced Pixel-AUC by 1.4% on MVTec-AD and 1.2% on VisA, while removing the multi-scale decoder led to a smaller but consistent performance decline. The complete removal of all multi-scale components (a single-scale SAM) resulted in the largest performance drop, particularly on data sets with large defect size variations such as VisA. This demonstrates that the multi-scale design enhances the model's robustness to defects of varying sizes and improves boundary accuracy.

Table 9: Comparison between SAM-LLaVA and its variant without the SASA module in Image-AUC, Pixel-AUC and accuracy (Acc.) obtained on the MVTec-AD [19] and VisA [20] data sets.

Model Variant	MVTec-AD			VisA		
	Image-AUC	Pixel-AUC	Acc.	Image-AUC	Pixel-AUC	Acc.
SAM-LLaVA	**95.3**	**96.1**	**92.3**	**88.2**	**96.5**	**83.8**
w/o SASA Module	94.5	95.3	89.6	87.2	95.4	80.1

Table 10: Comparison of different prompt encoding strategies used for SAM-LLaVA in Image-AUC, Pixel-AUC and accuracy (Acc.) obtained on the MVTec-AD [19] and VisA [20] data sets.

Prompt Type	MVTec-AD			VisA		
	Image-AUC	Pixel-AUC	Acc.	Image-AUC	Pixel-AUC	Acc.
Mask Prompt (Ours)	**95.3**	**96.1**	**92.3**	**88.2**	**96.5**	**83.8**
Box Prompt	94.2	94.8	91.5	86.5	95.0	82.5
Point Prompt	93.5	94.0	90.0	85.8	94.2	80.8

5.3.3. Impact of the Segmentation-Aware Semantic Alignment (SASA) Module

We evaluated the SASA module by replacing it with a simple concatenation of visual mask features and text embeddings (w/o SASA). As reported in Table 9, while the removal of SASA had a relatively minor impact on detection metrics (Pixel-AUC dropped by only 0.8% on MVTec-AD), it severely degraded the quality of generated descriptions. The accuracy of defect descriptions decreased by 2.7% on MVTec-AD and 3.6% on VisA. This indicates that SASA is essential for establishing fine-grained alignment between segmentation masks and textual concepts, thereby reducing hallucination and improving descriptive detail.

5.3.4. Comparison of Prompt Encoding Strategies

We compared three types of prompts used for SAM, including mask prompts (ours), box prompts and point prompts. As summarized in Table 10, mask prompts achieved the highest Pixel-AUC and description accuracy, outperforming box prompts by 1.3% in Pixel-AUC on MVTec-AD. Point prompts, while computationally lighter, resulted in significantly worse performance due to insufficient spatial guidance. This suggests that mask-level coarse localization provides richer contextual information for SAM to

Table 11: Comparison between three state-of-the-art LVLM-based methods and our SAM-LLaVA in computational complexity, including the number of total parameters, number of trainable parameters, total training time used on the TADD [12] data set, and average inference latency.

Method	Total Param. (M)	Trainable Param. (M)	Training Time (Hours)	Latency (ms)
AnomalyGPT [14]	8,089.64	150.44	41.2	1,253.05
Myriad [15]	8,321.62	231.50	50.5	1,809.70
SPGDD-GPT [12]	8,195.65	350.49	73.8	1,293.98
SAM-LLaVA (Ours)	8,334.92	112.38	35.0	1,563.86

perform precise segmentation.

5.3.5. Computational Complexity Analysis

To evaluate the practical deployability of SAM-LLaVA, we compared its computational complexity with three LVLM-based baselines, including AnomalyGPT [14], Myriad [15], and SPGDD-GPT [12]. All inference measurements were conducted on a single NVIDIA L40 GPU with a batch size of 1 and the images were resized to a resolution of 240×240 pixels. Latency was averaged over 200 runs after 50 warm-up iterations. As shown in Table 11, SAM-LLaVA contains 8,334.92M parameters in total, of which only 112.38M (1.35%) are trainable. But LoRA only introduces 31.59M trainable parameters (0.38% of the total parameters). The inference latency is 1,563.86ms and the throughput is 0.64 images/s, with a peak GPU memory usage of 21GB. The training operation performed on two NVIDIA L40 GPUs (with a per-GPU batch size of 4) consumed 28GB per GPU and took around 35.0 hours.

Therefore, SAM-LLaVA achieves a favorable balance between accuracy and efficiency, making it suitable for industrial deployment scenarios where diagnostic precision is prioritized over real-time speed.

6. Conclusion

In this paper, we proposed a Segmentation-Aware vision-language model, SAM-LLaVA, for general industrial defect diagnosis. To address the limitations of previous

methods in fine-grained segmentation and semantic alignment, we introduced three core innovations, including a CLIP-SAM cascade prompting mechanism, a multi-scale prompt learner and mask decoder, and a Segmentation-Aware Semantic Alignment (SASA) module. The CLIP-SAM cascade leverages the zero-shot capability of CLIP to generate coarse defect localization, which then guides SAM to perform precise pixel-level segmentation, effectively overcoming the patch-size limitation of CLIP-based methods. The multi-scale components enhance the model's adaptability to defects of varying sizes, while the SASA module establishes explicit fine-grained alignment between segmentation masks and textual concepts, significantly reducing description hallucination and improving semantic consistency.

Extensive experiments on multiple industrial defect data sets, including MVTec-AD, VisA and Text-Augmented Defect Data Set (TADD), demonstrated that SAM-LLaVA achieved state-of-the-art performance in both defect detection and description tasks. Under zero-shot and few-shot settings, SAM-LLaVA consistently outperformed existing methods in terms of Pixel-AUC, Image-AUC, and description accuracy. Ablation studies further validated the necessity of each proposed component, confirming that the CLIP-SAM cascade is crucial for accurate defect localization, the multi-scale design improves robustness to scale variation, and the SASA module is essential for generating detailed and faithful textual descriptions.

CRediT authorship contribution statement

Shengwang An: Data curation, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft; **Chengjia Wang:** Formal analysis, Methodology, Writing - Review & Editing; **Xinghui Dong:** Conceptualization, Funding acquisition, Investigation, Methodology, Project Administration, Resources, Supervision, Writing - Review & Editing.

Acknowledgement

This study was supported by the National Natural Science Foundation of China (NSFC) (No. 42576200).

References

[1] J. Guo, G. Song, and Y. Wang, "A memory and retrieval transformer-based unsupervised learning model for anomaly detection and segmentation," *Pattern Recognition*, vol. 174, p. 113004, 2026.

[2] B. Yang, Z. Liu, G. Duan, and J. Tan, "Residual shape adaptive dense-nested Unet: Redesign the long lateral skip connections for metal surface tiny defect inspection," *Pattern Recognition*, vol. 147, p. 110073, 2024.

[3] Y. Kim, "Convolutional Neural Networks for Sentence Classification," *arXiv e-prints*, p. arXiv:1408.5882, Aug. 2014.

[4] A. Dosovitskiy, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[5] K. Roth, L. Pemula, J. Zepeda, B. Scholkopf, T. Brox, and P. Gehler, "Towards Total Recall in Industrial Anomaly Detection," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14 298–14 308, 2021.

[6] N. Cohen and Y. Hoshen, "Sub-Image Anomaly Detection with Deep Pyramid Correspondences," *ArXiv*, vol. abs/2005.02357, 2020.

[7] M. Yang, P. Wu, J. Liu, and H. Feng, "MemSeg: A semi-supervised method for image surface defect detection using differences and commonalities," *Eng. Appl. Artif. Intell.*, vol. 119, p. 105835, 2022.

[8] C. Huang, H. Guan, A. Jiang, Y. Zhang, M. W. Spratling, and Y. Wang, "Registration based Few-Shot Anomaly Detection," *ArXiv*, vol. abs/2207.07361, 2022.

[9] J. Zhu and G. Pang, "Toward Generalist Anomaly Detection via In-Context Residual Learning with Few-Shot Sample Prompts," *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17 826–17 836, 2024.

[10] Y. Li, H. Wang, Y. Duan, and X. Li, "A closer look at the explainability of contrastive language-image pre-training," *Pattern Recognit.*, vol. 162, p. 111409, 2023.

[11] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, 2021.

[12] S. An and X. Dong, "SPGDD-GPT: Image-Text-Driven Generic Defect Diagnosis Using a Self-prompted Large Vision-Language Model," *IEEE Transactions on Automation Science and Engineering*, pp. 1–1, 2026.

[13] Z. Qu, X. Tao, X. Gong, S. Qu, Q. Chen, Z. Zhang, X. Wang, and G. Ding, "Bayesian Prompt Flow Learning for Zero-Shot Anomaly Detection," 2025.

[14] Z. Gu, B. Zhu, G. Zhu, Y. Chen, M. Tang, and J. Wang, "AnomalyGPT: Detecting Industrial Anomalies using Large Vision-Language Models," in *AAAI Conference on Artificial Intelligence*, 2023.

[15] Y. Li, H. Wang, S. Yuan, M. Liu, D. Zhao, Y. Guo, C. Xu, G. Shi, and W. Zuo, "Myriad: Large multimodal model by applying vision experts for industrial anomaly detection," *arXiv preprint arXiv:2310.19070*, 2023.

[16] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. B. Girshick, "Segment anything," *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3992–4003, 2023.

[17] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual Instruction Tuning," *ArXiv*, vol. abs/2304.08485, 2023.

[18] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," *ArXiv*, vol. abs/2106.09685, 2021.

[19] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9584–9592.

[20] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer, "SPot-the-Difference Self-Supervised Pre-training for Anomaly Detection and Segmentation," *arXiv preprint arXiv:2207.14315*, 2022.

[21] Y. Li, Y. Cao, C. Liu, Y. Xiong, X. Dong, and C. Huang, "IAD-R1: Reinforcing Consistent Reasoning in Industrial Anomaly Detection," *ArXiv*, vol. abs/2508.09178, 2025.

[22] Y. Chengxiao, J. Owais, A. Ahmed, M. O. Khan, Z. Xiaoyang, and M. H. Tunio, "Software Defect Prediction Using Machine Learning - A Systematic Literature Review," in *2023 20th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2023, pp. 1–9.

[23] J. Jeong, Y. Zou, T. Kim, D. Zhang, A. Ravichandran, and O. Dabeer, "WinCLIP: Zero-/Few-Shot Anomaly Classification and Segmentation," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19 606–19 616, 2023.

[24] Y. Cao, X. Xu, C. Sun, Y. Cheng, Z. Du, L. Gao, and W. Shen, "Personalizing Vision-Language Models With Hybrid Prompts for Zero-Shot Anomaly Detection," *IEEE Transactions on Cybernetics*, vol. 55, pp. 1917–1929, 2023.

[25] Y. Huang, J. Zhu, X. Zhong, and Y. Deng, "SAID: Segment All Industrial Defects with Scene Prompts," *Sensors (Basel, Switzerland)*, vol. 25, 2025.

[26] S. Li, J. Cao, P. Ye, Y. Ding, C. Tu, and T. Chen, "ClipSAM: CLIP and SAM Collaboration for Zero-Shot Anomaly Segmentation," *ArXiv*, vol. abs/2401.12665, 2024.

[27] Y. Peng, X. Lin, N. Ma, J. Du, C. Liu, C. Liu, and Q. Chen, "SAM-LAD: Segment Anything Model Meets Zero-Shot Logic Anomaly Detection," 2025.

[28] Z. Lu, C. Sun, and X. Li, "Multimodal Large Language Model-Enabled Machine Intelligent Fault Diagnosis Method with Non-Contact Dynamic Vision Data," *Sensors*, vol. 25, no. 18, 2025.

[29] S. Bai, Y. Cai, R. Chen, K. Chen, X.-H. Chen, Z. Cheng, L. Deng, W. Ding, R. Fang, C. Gao, C. Ge, W. Ge, Z. Guo, Q. Huang, Q. Huang, F. Huang, B. Hui, S. Jiang, Z. Li, M. Li, M. Li, K. Li, Z. Lin, J. Lin, X. Liu, J. Liu, C. Liu, Y. Liu, D. Liu, S. Liu, D. Lu, R. Luo, C. Lv, R. Men, L. Y. Meng, X. Ren, X. yi Ren, S. Song, Y.-C. Sun, J. Tang, J. Tu, J. Wan, P. Wang, P. Wang, Q. Wang, Y. Wang, T. Xie, Y. Xu, H. Xu, J. Xu, Z. Yang, M. Yang, J. Yang, A. Yang, B. Yu, F. Zhang, H. Zhang, X. Zhang, B. Zheng, H. Zhong, J. Zhou, F. Zhou, J. Zhou, Y. Zhu, and K. Zhu, "Qwen3-VL Technical Report," *ArXiv*, 2025.

[30] Y. Su, T. Lan, H. Li, J. Xu, Y. Wang, and D. Cai, "PandaGPT: One Model To Instruction-Follow Them All," *ArXiv*, vol. abs/2305.16355, 2023.

[31] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez *et al.*, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," vol. 2, no. 3, p. 6, 2023.

[32] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[33] W. Ma, X. Zhang, Q. Yao, F. Tang, C. Wu, Y. Li, R. Yan, Z. Jiang, and S. Zhou, "AA-CLIP: Enhancing Zero-Shot Anomaly Detection via Anomaly-Aware CLIP," *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4744–4754, 2025.

[34] Y. Cao, J. Zhang, L. Frittoli, Y. Cheng, W. Shen, and G. Boracchi, "AdaCLIP: Adapting CLIP with Hybrid Learnable Prompts for Zero-Shot Anomaly Detection," *ArXiv*, vol. abs/2407.15795, 2024.

[35] Q. Zhou, G. Pang, Y. Tian, S. He, and J. Chen, "AnomalyCLIP: Object-agnostic Prompt Learning for Zero-shot Anomaly Detection," *ArXiv*, vol. abs/2310.18961, 2023.

[36] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection and Localization," *arXiv e-prints*, p. arXiv:2011.08785, Nov. 2020.

[37] X. Li, Z. Zhang, X. Tan, C. Chen, Y. Qu, Y. Xie, and L. Ma, "PromptAD: Learning Prompts with only Normal Samples for Few-Shot Anomaly Detection," *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16 848–16 858, 2024.

[38] X. Yao, Y. Luo, Z. Qian, and C. Zhang, "ADPretrain: Advancing Industrial Anomaly Detection via Anomaly Representation Pretraining," *ArXiv*, vol. abs/2511.05245, 2025.

[39] C. Fu, H. Fang, X. Zheng, W. Wei, Y. Li, H. Sun, and X. Li, "SSVP: Synergistic Semantic-Visual Prompting for Industrial Zero-Shot Anomaly Detection," *ArXiv*, vol. abs/2601.09147, 2026.

[40] Y. Li, H. Fan, R. Hu, C. Feichtenhofer, and K. He, "Scaling language-image pre-training via masking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 390–23 400.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

[42] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny, "MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models," *ArXiv*, vol. abs/2304.10592, 2023.

[43] Z. Yang, L. Li, K. Lin, J. Wang, C.-C. Lin, Z. Liu, and L. Wang, "The Dawn of LMMs: Preliminary Explorations with GPT-4V(ision)," *ArXiv*, vol. abs/2309.17421, 2023.

[44] X. Zhang, Y. Lu, W. Wang, A. Yan, J. Yan, L. Qin, H. Wang, X. Yan, W. Y. Wang, and L. R. Petzold, "GPT-4V(ision) as a Generalist Evaluator for Vision-Language Tasks," *ArXiv*, vol. abs/2311.01361, 2023.

[45] K. Maeda, S. Kurita, T. Miyanishi, and N. Okazaki, "Vision Language Model-based Caption Evaluation Method Leveraging Visual Context Extraction," *ArXiv*, vol. abs/2402.17969, 2024.