AnomalyGPT: Detecting Industrial Anomalies using Large Vision-Language Models

Zhaopeng Gu[1,2] Bingke Zhu[1,3,4] Guibo Zhu[1,2,4]
Yingying Chen[1,3,4] Ming Tang[1,2] Jinqiao Wang[1,2,3,4]
[1] Foundation Model Research Center, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Objecteye Inc., Beijing, China
[4] Wuhan AI Research, Wuhan, China
guzhaopeng2023@ia.ac.cn
{bingke.zhu,gbzhu,yingying.chen,tangm,jqwang}@nlpr.ia.ac.cn

Abstract

Large Vision-Language Models (LVLMs) such as MiniGPT-4 and LLaVA have demonstrated the capability of understanding images and achieved remarkable performance in various visual tasks. Despite their strong abilities in recognizing common objects due to extensive training datasets, they lack specific domain knowledge and have a weaker understanding of localized details within objects, which hinders their effectiveness in the Industrial Anomaly Detection (IAD) task. On the other hand, most existing IAD methods only provide anomaly scores and necessitate the manual setting of thresholds to distinguish between normal and abnormal samples, which restricts their practical implementation. In this paper, we explore the utilization of LVLM to address the IAD problem and propose AnomalyGPT, a novel IAD approach based on LVLM. We generate training data by simulating anomalous images and producing corresponding textual descriptions for each image. We also employ an image decoder to provide fine-grained semantic and design a prompt learner to fine-tune the LVLM using prompt embeddings. Our AnomalyGPT eliminates the need for manual threshold adjustments, thus directly assesses the presence and locations of anomalies. Additionally, AnomalyGPT supports multi-turn dialogues and exhibits impressive few-shot in-context learning capabilities. With only one normal shot, AnomalyGPT achieves the state-of-the-art performance with an accuracy of 86.1%, an image-level AUC of 94.1%, and a pixel-level AUC of 95.3% on the MVTec-AD dataset. Code is available at https://github.com/CASIA-IVA-Lab/AnomalyGPT.

1. Introduction

Large Language Models (LLMs) like GPT-3.5 [19] and LLaMA [26] have demonstrated remarkable performance on a range of Natural Language Processing (NLP) tasks. More recently, novel methods including MiniGPT-4 [36], BLIP-2 [15], and PandaGPT [25] have further extended the ability of LLMs into visual processing by aligning visual features with text features, bringing a significant revolution in the domain of Artificial General Intelligence (AGI). While LVLMs are pre-trained on amounts of data sourced from the Internet, their domain-specific knowledge is relatively limited and they lack sensitivity to local details within objects, which restricts their potentiality in IAD task.

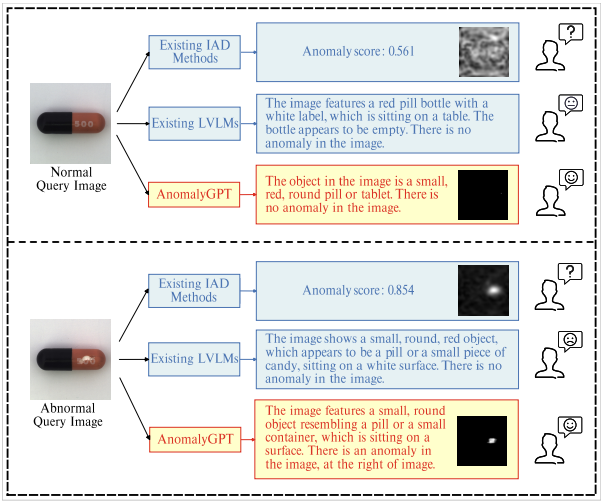

Figure 1. Comparison between our AnomalyGPT, existing IAD methods and existing LVLMs. Existing IAD methods can only provide anomaly scores and need manually threshold setting, while existing LVLMs cannot detect anomalies in the image. AnomalyGPT can not only provide information about the image but also indicate the presence and location of anomaly.

IAD task aims to detect and localize anomalies in in-

Methods	Few-shot learning	Anomaly score	Anomaly localization	Anomaly judgement	Multi-turn dialogue
Traditional IAD methods		✓	✓		
Few-shot IAD methods	✓	✓	✓		
LVLMs	✓				✓
AnomalyGPT (ours)	✓	✓	✓	✓	✓

Table 1. Comparison between our AnomalyGPT and existing methods across various functionalities. The "Traditional IAD methods" in the table refers to "one-class-one-model" methods such as PatchCore [23], InTra [21], and PyramidFlow [13]. "Few-shot IAD methods" refers to methods that can perform few-shot learning like RegAD [10], Graphcore [29], and WinCLIP [27]. "LVLMs" represents general large vision-language models like MiniGPT-4 [36], LLaVA [17], and PandaGPT [25]. "Anomaly score" in the table represents just providing scores for anomaly detection, while "Anomaly judgement" indicates directly assessing the presence of anomaly.

dustrial product images. Due to the rarity and unpredictability of real-world samples, models are required to be trained only on normal samples and distinguish anomalous samples that deviate from normal samples. Current IAD methods [10, 11, 32] typically only provide anomaly scores for test samples and require manually specification of thresholds to distinguish between normal and anomalous instances for each class of items, which is not suitable for real production environments.

As illustrated in Figure 1 and Table 1, neither existing IAD methods nor LVLMs can address IAD problem well, so we introduce AnomalyGPT, a novel IAD approach based on LVLM. AnomalyGPT can detect the presence and location of anomalies without the need for manual threshold settings. Moreover, our method can provide information about the image and allows for interactive engagement, enabling users to ask follow-up questions based on their needs and the provided answers. AnomalyGPT can also perform in-context learning with a small number of normal samples, enabling swift adaptation to previously unseen objects.

Specifically, we focus on fine-tuning the LVLM using synthesized anomalous visual-textual data, integrating IAD knowledge into the model. However, direct training with IAD data presents numerous challenges. The first is data scarcity. Methods like LLaVA [17] and PandaGPT [25] are pre-trained on 160k images with corresponding multi-turn dialogues. However, existing IAD datasets [1, 37] contain only a few thousand samples, rendering direct fine-tuning easy to overfitting and catastrophic forgetting. To address this, we use prompt embeddings to fine-tune the LVLM instead of parameter fine-tuning. Additional prompt embeddings are added after image inputs, introducing supplementary IAD knowledge into the LVLM. The second challenge relates to fine-grained semantic. We propose a lightweight, visual-textual feature-matching-based decoder to generate pixel-level anomaly localization results. The decoder's outputs are introduced to the LVLM along with the original test images through prompt embeddings, which allows the LVLM to utilize both the raw image and the decoder's outputs to make anomaly determinations, improving the accuracy of its judgments.

Experimentally, we conduct extensive experiments on the MVTec-AD [1] and VisA [37] datasets. With unsupervised training on the MVTec-AD dataset, we achieve an accuracy of 93.3%, an image-level AUC of 97.4%, and a pixel-level AUC of 93.1%. When one-shot transferred to the VisA dataset, we achieve an accuracy of 77.4%, an image-level AUC of 87.4%, and a pixel-level AUC of 96.2%. Conversely, after unsupervised training on the VisA dataset, one-shot transferred to the MVTec-AD dataset result in an accuracy of 86.1%, an image-level AUC of 94.1%, and a pixel-level AUC of 95.3%.

Our contributions are summarized as follows:

- We present the pioneering utilization of LVLM for addressing IAD task. Our method not only detects and locates anomaly without manually threshold adjustments but also supports multi-round dialogues. To the best of our knowledge, we are the first to successfully apply LVLM to the domain of industrial anomaly detection.
- The lightweight, visual-textual feature-matching-based decoder in our work addresses the limitation of the LLM's weaker discernment of fine-grained semantic and alleviates the constraint of LLM's restricted ability to solely generate text outputs.
- We employ prompt embeddings for fine-tuning and train our model concurrently with the data utilized during LVLM pre-training, thus preserving the LVLM's inherent capabilities and enabling multi-turn dialogues.
- Our method retains robust transferability and is capable of engaging in in-context few-shot learning on new datasets, yielding outstanding performance.

2. Related Work

Industial Anomaly Detection: Existing IAD methods can be categorized into reconstruction-based and feature embedding-based approaches. Reconstruction-based methods primarily aim to reconstruct anomalous samples to their corresponding normal counterparts and detect anomalies by calculating the reconstruction error. RIAD [33], SCADN [30], InTra [21] and AnoDDPM [28] employ different reconstruction network architectures, ranging from

autoencoder and Generative Adversarial Network (GAN) to Transformer and diffusion model.

Feature embedding-based methods focus on modeling the feature embeddings of normal samples. Approaches such as PatchSVDD [31] aim to find a hypersphere that tightly encapsulates normal samples. Cflow-AD [9] and PyramidFlow [13] use normalizing flows to project normal samples onto a Gaussian distribution. PatchCore [23] and CFA [12] establish a memory bank of patch embeddings from normal samples and detect anomalies by measuring the distance between a test sample embedding and its nearest normal embedding in the memory bank.

These methods typically follow the "one-class-one-model" learning paradigm, requiring plentiful normal samples for each object class to learn its distribution, making them impractical for novel object categories and less suitable for dynamic production environments. In contrast, our method facilitates in-context learning for novel object categories, enabling inference with only few normal samples.

Zero-/Few-shot Industial Anomaly Detection: Recent efforts have focused on methods utilizing minimal normal samples to accomplish IAD task. PatchCore [23] constructs a memory bank using only a few normal samples, resulting in a noticeable performance decline. RegAD [10] trained an image registration network to align test images with normal samples, followed by similarity computation for corresponding patches. WinCLIP [11] leveraged CLIP [22] to compute similarity between images and textual descriptions representing normal and anomalous semantics, distinguishing anomalies based on their relative scores. However, these methods can only provide anomaly scores for test samples during inference. To distinguish normal samples from anomalous ones, it's necessary to experimentally determine the optimal threshold on a test set, which contradicts the original intent of IAD task that only utilize normal data. For instance, while PatchCore [23] achieves an image-level AUC of 99.3% on MVTec-AD in unsupervised setting, its accuracy drops to 79.76% when using a unified threshold for inference. The detailed experimental results and analyses can be found in Appendix A. Our method, in contrast, enables the LVLM to directly assess test samples for the presence of anomalies and pinpoint their locations, demonstrating enhanced practicality.

Large Vision-Language Models: LLMs, traditionally successful in NLP, are now explored for visual tasks. BLIP-2 [15] leverages Q-Former to input visual features from Vision Transformer [7] into the Flan-T5 [4] model. MiniGPT-4 [36] connects the image segment of BLIP-2 and the Vicuna [3] model with a linear layer, performing a two-stage fine-tuning process using extensive image-text data. PandaGPT [25] establishes a connection between ImageBind [8] and the Vicuna [3] model via a linear layer, allowing for multi-modal input. These approaches showcase the potential of LLM-based polymathic models.

However, as mentioned earlier, these models are trained on general data and lack domain-specific expertise. In this paper, through the utilization of simulated anomaly data, image decoder and prompt embeddings, AnomalyGPT is introduced as an novel approach that achieves IAD task without the need for manually specified thresholds, while also enabling few-shot in-context learning. Table 1 illustrates a comparison between AnomalyGPT and existing methods across various functionalities.

3. Method

AnomalyGPT is a novel conversational IAD vision-language model, primarily designed for detecting anomalies in images of industrial artifacts and pinpointing their positions. We leverage a pre-trained image encoder and a LLM to align IAD images and their corresponding textual descriptions via simulated anomaly data. We introduce a decoder module and a prompt learner module to enhance IAD performance and achieve pixel-level localization output. Employing prompt tuning and alternate training with pre-training data preserves the LLM's transferability and prevents catastrophic forgetting. Our method exhibits robust few-shot transfer capability, enabling anomaly detection and localization for previously unseen items with merely one normal sample provided.

3.1. Model architecture

Figure 2 illustrates the comprehensive architecture of AnomalyGPT. Given a query image $x \in \mathbb{R}^{H\times W\times C}$, the final features $F_{img} \in \mathbb{R}^{C_1}$ extracted by the image encoder are passed through the linear layer to obtain the image embedding $E_{img} \in \mathbb{R}^{C_{emb}}$, which is then fed into the LLM. In unsupervised setting, the patch-level features extracted by intermediate layers of image encoder are fed into the decoder together with text features to generate pixel-level anomaly localization results. In few-shot setting, the patch-level features from normal samples are stored in memory banks and the localization result can be obtained by calculating the distance between query patches and their most similar counterparts in the memory bank. The localization results is subsequently transformed into prompt embeddings through the prompt learner, serving as a part of LLM input. The LLM leverages image input, prompt embeddings, and user-provided textual input to detect anomalies and identify their locations, thus generating responses for the user.

3.2. Decoder and prompt learner

Decoder To achieve pixel-level anomaly localization, we employ a lightweight feature-matching-based image decoder that supports both unsupervised IAD and few-shot IAD. The design of the decoder is primarily inspired by PatchCore [23], WinCLIP [11], and APRIL-GAN [2].

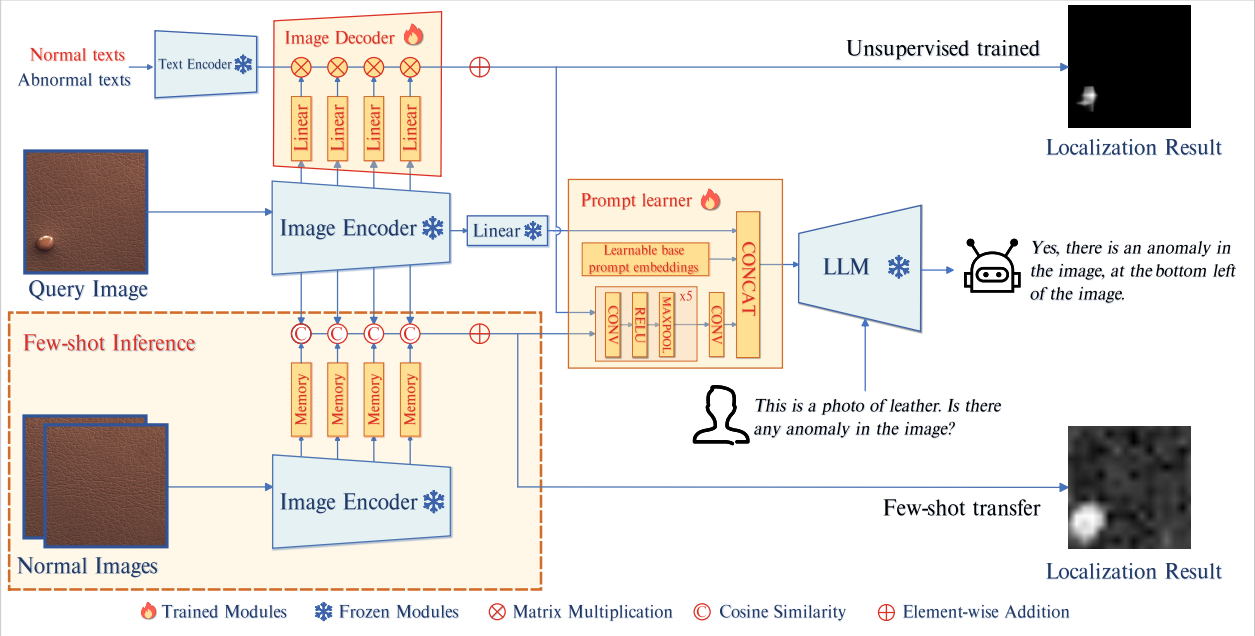

Figure 2. The architecture of AnomalyGPT. The query image is passed to the frozen image encoder and the patch-level features extracted from intermediate layers are fed into image decoder to compute their similarity with normal and abnormal texts to obtain localization result. The final features extracted by the image encoder are fed to a linear layer and then passed to the prompt learner along with the localization result. The prompt learner converts them into prompt embeddings suitable for input into the LLM together with user text inputs. In few-shot setting, the patch-level features from normal samples are stored in memory banks and the localization result can be obtained by calculating the distance between query patches and their most similar counterparts in the memory bank.

As illustrated in the upper part of Figure 2, we partition the image encoder into 4 stages and obtain the intermediate patch-level features extracted by every stage $F^i_{patch} \in \mathbb{R}^{H_i \times W_i \times C_i}$, where i indicates the i-th stage. Following the idea from WinCLIP [11], a natural approach is to compute the similarity between F^i_{patch} and the text features $F_{text} \in \mathbb{R}^{2 \times C_{text}}$ respectively representing normality and abnormality. Detailed texts representing normal and abnormal cases are presented in Appendix B. However, since these intermediate features have not undergone the final image-text alignment, they cannot be directly compared with text features. To address this, we introduce additional linear layers to project these intermediate features to $\tilde{F}^i_{patch} \in \mathbb{R}^{H_i \times W_i \times C_{text}}$, and align them with text features representing normal and abnormal semantics. The localization result $M \in \mathbb{R}^{H \times W}$ can be obtained by Eq. (1):

$$M = Upsample\left(\sum_{i=1}^{4} softmax(\tilde{F}^i_{patch} F^T_{text})\right). \quad (1)$$

For few-shot IAD, as illustrated in the lower part of Figure 2, we utilize the same image encoder to extract intermediate patch-level features from normal samples and store them in memory banks $B^i \in \mathbb{R}^{N \times C_i}$, where i indicates the i-th stage. For patch-level features $F^i_{patch} \in \mathbb{R}^{H_i \times W_i \times C_i}$, we calculate the distance between each patch and its most similar counterpart in the memory bank, and the localization result $M \in \mathbb{R}^{H \times W}$ can be obtained by Eq. (2):

$$M = Upsample\left(\sum_{i=1}^{4} \left(1 - max(F^i_{patch} \cdot {B^i}^T)\right)\right). \quad (2)$$

Prompt learner To leverage fine-grained semantic from images and maintain semantic consistency between LLM and decoder outputs, we introduce a prompt learner that transforms the localization result into prompt embeddings. Additionally, learnable base prompt embeddings, unrelated to decoder outputs, are incorporated into the prompt learner to provide extra information for the IAD task. Finally, these embeddings, along with the original image information, are fed into the LLM.

As illustrated in Figure 2, the prompt learner consists of the learnable base prompt embeddings $E_{base} \in \mathbb{R}^{n_1 \times C_{emb}}$ and a convolutional neural network. The network converts the localization result $M \in R^{H \times W}$ into n_2 prompt embeddings $E_{dec} \in \mathbb{R}^{n_2 \times C_{emb}}$. E_{base} and E_{dec} form a set of $n_1 + n_2$ prompt embeddings $E_{prompt} \in \mathbb{R}^{(n_1+n_2) \times C_{emb}}$ that are combined with the image embedding into the LLM.

3.3. Data for image-text alignment

Anomaly Simulation We primarily adopt the approach proposed by NSA [24] to simulate anomalous data. The NSA [24] method builds upon the Cut-paste [14] technique by incorporating the Poisson image editing [20] method to alleviate the discontinuity introduced by pasting image segments. Cut-paste [14] is a common technique in IAD domain for generating simulated anomaly images. This method involves randomly cropping a block region from an image and then pasting it onto a random location in another image, thus creating a simulated anomalous portion. Simulated anomaly samples can significantly enhance the performance of IAD models, but this procedure often results in noticeable discontinuities, as illustrated in Figure 3. The Poisson editing method [20] has been developed to seamlessly clone an object from one image into another image by solving the Poisson partial differential equations.

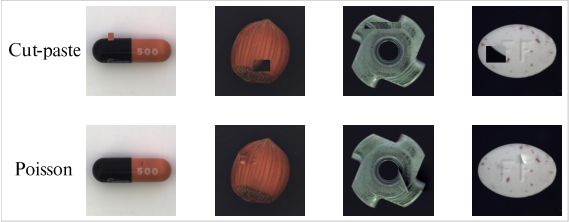

Figure 3. Illustration of the comparison between cut-paste and poisson image editing. The results of cut-paste exhibit evident discontinuities and the results of poisson image editing are more natural.

Question and Answer Content To conduct prompt tuning on the LVLM, we generate corresponding textual queries based on the simulated anomalous images. Specifically, each query consists of two components. The first part involves a description of the input image, providing information about the objects present in the image and their expected attributes, such as *This is a photo of leather, which should be brown and without any damage, flaw, defect, scratch, hole or broken part.* The second part queries the presence of anomalies within the object, namely *Is there any anomaly in the image?* The LVLM firstly responds to whether anomalies are present. If anomalies are detected, the model continues to specify the number and location of the anomalous areas, such as *Yes, there is an anomaly in the image, at the bottom left of the image.* or *No, there are no anomalies in the image.* We divide the image into a grid of 3×3 distinct regions to facilitate the LVLM in verbally indicating the positions of anomalies, as shown in Figure 4. The descriptive content about the image furnishes the LVLM with foundational knowledge of the input image, aiding in the model's better comprehension of the image contents. However, during practical applications, users may opt to omit this descriptive input, and the model is still capable of performing IAD task based solely on the provided image input. Detailed description for each category are provided in Appendix C.

Figure 4. Illustration of the 3×3 grid of image, which is used to let LLM verbally indicate the abnormal position.

Prompts fed to the LLM typically follow the format:

Human: <Img>E_{img}*</Img>*E_{prompt}*[Image Description]Is there any anomaly in the image?###Assistant:*

$E_{img} \in \mathbb{R}^{C_{emb}}$ represents the image embedding being processed through the image encoder and linear layer, $E_{prompt} \in \mathbb{R}^{(n_1+n_2)\times C_{emb}}$ refers to the prompt embeddings generated by the prompt learner, and *[Image Description]* corresponds to the textual description of the image.

3.4. Loss Functions

To train the decoder and prompt learner, we primarily employed three loss functions: cross-entropy loss, focal loss [16], and dice loss [18]. The latter two are primarily utilized to enhance the pixel-level localization accuracy of the decoder.

Cross-entropy Loss Cross-entropy loss is commonly employed for training language models, which quantifies the disparity between the text sequence generated by the model and the target text sequence. The formula is as follows:

$$L_{ce} = -\sum_{i=1}^{n} y_i log(p_i), \tag{3}$$

where n is the number of tokens, y_i is the true label for token i and p_i is the predicted probability for token i.

Focal Loss Focal loss [16] is commonly used in object detection and semantic segmentation to address the issue of class imbalance, which introduces an adjustable parameter γ to modify the weight distribution of cross-entropy loss, emphasizing samples that are difficult to classify. In IAD task, where most regions in anomaly images are still normal, employing focal loss can mitigate the problem of class imbalance. Focal loss can be calculated by Eq. (4):

$$L_{focal} = -\frac{1}{n}\sum_{i=1}^{n} (1-p_i)^{\gamma} log(p_i), \tag{4}$$

Setup	Method	MVTec-AD			VisA		
		Image-AUC	Pixel-AUC	Accuracy	Image-AUC	Pixel-AUC	Accuracy
1-shot	SPADE	81.0 ± 2.0	91.2 ± 0.4	-	79.5 ± 4.0	95.6 ± 0.4	-
	PaDiM	76.6 ± 3.1	89.3 ± 0.9	-	62.8 ± 5.4	89.9 ± 0.8	-
	PatchCore	83.4 ± 3.0	92.0 ± 1.0	-	79.9 ± 2.9	95.4 ± 0.6	-
	WinCLIP	93.1 ± 2.0	95.2 ± 0.5	-	83.8 ± 4.0	**96.4 ± 0.4**	-
	AnomalyGPT (ours)	**94.1 ± 1.1**	**95.3 ± 0.1**	**86.1 ± 1.1**	**87.4 ± 0.8**	96.2 ± 0.1	**77.4 ± 1.0**
2-shot	SPADE	82.9 ± 2.6	92.0 ± 0.3	-	80.7 ± 5.0	96.2 ± 0.4	-
	PaDiM	78.9 ± 3.1	91.3 ± 0.7	-	67.4 ± 5.1	92.0 ± 0.7	-
	PatchCore	86.3 ± 3.3	93.3 ± 0.6	-	81.6 ± 4.0	96.1 ± 0.5	-
	WinCLIP	94.4 ± 1.3	**96.0 ± 0.3**	-	84.6 ± 2.4	**96.8 ± 0.3**	-
	AnomalyGPT (ours)	**95.5 ± 0.8**	95.6 ± 0.2	**84.8 ± 0.8**	**88.6 ± 0.7**	96.4 ± 0.1	**77.5 ± 0.3**
4-shot	SPADE	84.8 ± 2.5	92.7 ± 0.3	-	81.7 ± 3.4	96.6 ± 0.3	-
	PaDiM	80.4 ± 2.5	92.6 ± 0.7	-	72.8 ± 2.9	93.2 ± 0.5	-
	PatchCore	88.8 ± 2.6	94.3 ± 0.5	-	85.3 ± 2.1	96.8 ± 0.3	-
	WinCLIP	95.2 ± 1.3	96.2 ± 0.3	-	87.3 ± 1.8	**97.2 ± 0.2**	-
	AnomalyGPT (ours)	**96.3 ± 0.3**	**96.2 ± 0.1**	**85.0 ± 0.3**	**90.6 ± 0.7**	96.7 ± 0.1	**77.7 ± 0.4**

Table 2. Few-shot IAD results on MVTec-AD and VisA datasets. Results are listed as the average of 5 runs and the best-performing method is in **bold**. The results for SPADE, PaDiM, PatchCore and WinCLIP are reported from [11].

Method	Image-AUC	Pixel-AUC	Accuracy
PaDiM (Unified)	84.2	89.5	-
JNLD (Unified)	91.3	88.6	-
UniAD	96.5	**96.8**	-
AnomalyGPT (ours)	**97.4**	93.1	**93.3**

Table 3. Unsupervised anomaly detection results on MVTec-AD dataset. The best-performing method is in **bold** and the results for PaDiM and JNLD are reported from [35].

where $n = H \times W$ represents the total number of pixels, p_i is the predicted probability of the positive classes and γ is a tunable parameter for adjusting the weight of hard-to-classify samples. In our implementation, we set γ to 2.

Dice Loss Dice loss [18] is a commonly employed loss function in semantic segmentation tasks. It is based on the dice coefficient and can be calculated by Eq. (5):

$$L_{dice} = -\frac{\sum_{i=1}^{n} y_i \hat{y}_i}{\sum_{i=1}^{n} y_i^2 + \sum_{i=1}^{n} \hat{y}_i^2}, \tag{5}$$

where $n = H \times W$, y_i is the output of decoder and $\hat{y}_i$ is the ground truth value.

Finally, the overall loss function is defined as:

$$L = \alpha L_{ce} + \beta L_{focal} + \delta L_{dice}, \tag{6}$$

where α, β, δ are coefficients to balance the three loss functions, which are set to 1 by default in our experiments.

4. Experiments

Datasets We conduct experiments primarily on the MVTec-AD [1] and VisA [37] datasets. The MVTec-AD dataset comprises 3629 training images and 1725 testing images across 15 different categories, making it one of the most popular datasets for IAD. The training images only consist of normal images, while the testing images contain both normal and anomalous images. The image resolutions vary from 700×700 to 1024×1024. VisA, a newly introduced IAD dataset, contains 9621 normal images and 1200 anomalous images across 12 categories, with resolutions approximately around 1500×1000. Consistent with previous IAD methods, we only use the normal data from these datasets for training.

Evaluation metrics Following existing IAD methods, we employ the Area Under the Receiver Operating Characteristic (AUC) as our evaluation metric, with image-level and pixel-level AUC used to assess anomaly detection and anomaly localization performance, respectively. However, our proposed approach uniquely allows for determining the presence of anomalies without the need for manually-set thresholds. Therefore, we also utilize the image-level accuracy to evaluate the performance of our method.

Implementation details We utilize ImageBind-Huge [8] as the image encoder and Vicuna-7B [3] as the inferential LLM, connected through a linear layer. We initialize our model using pre-trained parameters from PandaGPT [25]. We set the image resolution at 224×224 and feed the

Decoder	Prompt learner	LLM	LoRA	MVTec-AD (unsupervised)			VisA (1-shot)		
				Image-AUC	Pixel-AUC	Accuracy	Image-AUC	Pixel-AUC	Accuracy
		✓		-	-	72.2	-	-	56.5
	✓	✓		-	-	73.4	-	-	56.6
		✓	✓	-	-	79.8	-	-	63.4
✓		✓		97.1	90.9	72.2	85.8	96.2	56.5
✓		✓	✓	97.1	90.9	84.2	85.8	96.2	64.7
✓	✓	✓	✓	96.0	88.1	83.9	85.8	**96.5**	72.7
✓				97.1	90.9	90.3	85.8	96.2	75.4
✓	✓	✓		**97.4**	**93.1**	**93.3**	**87.4**	96.2	**77.4**

Table 4. Results of ablation studies. The ✓ in "Decoder" and "Prompt learner" columns indicate module inclusion. The ✓ in "LLM" column denotes whether use LLM for inference and the ✓ in "LoRA" column denotes whether use LoRA to fine-tune LLM. In settings without LLM, the maximum anomaly score from normal samples is used as the classification threshold. In settings without decoder, due to the sole textual output from the LLM, we cannot compute image-level and pixel-level AUC.

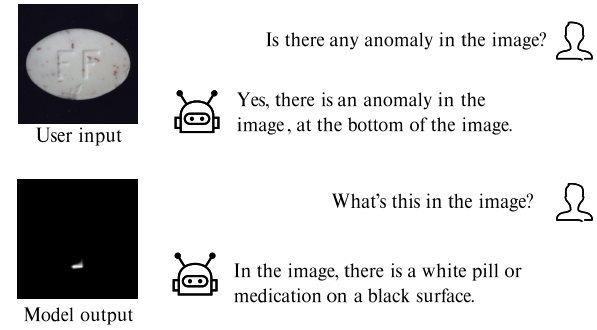

Figure 5. Qualitative example of AnomalyGPT in the unsupervised setting. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

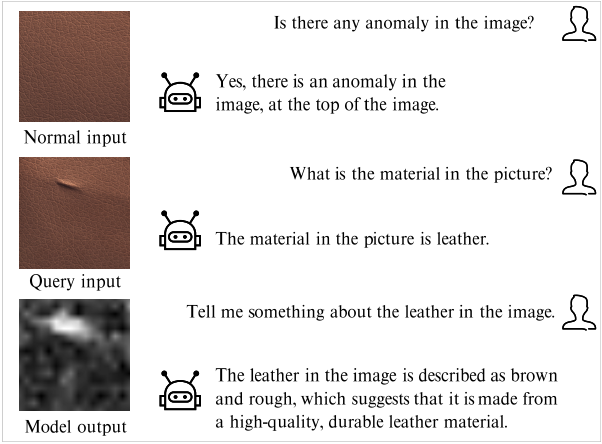

Figure 6. Qualitative example of AnomalyGPT in the **one-normal-shot** setting. The localization performance is slightly lower compared to the unsupervised setting due to the absence of parameter training.

outputs from the 8th, 16th, 24th, and 32nd layers of ImageBind-Huge's image encoder to the image decoder. Training is conducted on two RTX-3090 GPUs over 50 epochs, with a learning rate of 1e-3 and a batch size of 16. Linear warm-up and a one-cycle cosine learning rate decay strategy are applied. We perform alternating training using both the pre-training data of PandaGPT [25] and our anomaly image-text data. Only the decoder and prompt learner undergo parameter updates, while the remaining parameters are all kept frozen.

4.1. Quantitative Results

Few-shot Industial Anomaly Detection We compare our work with prior few-shot IAD methods, selecting SPADE [5], PaDiM [6], PatchCore [23], and WinCLIP [11] as the baselines. The results are presented in Table 2. Across both datasets, our method notably outperforms previous approaches in terms of image-level AUC and achieves competitive pixel-level AUC and good accuracy.

Unsupervised Industial Anomaly Detection In the setting of unsupervised training with a large number of normal samples, given that our method trains a single model on samples from all classes within a dataset, we selected UniAD [32], which is trained under the same setup, as a baseline for comparison. Additionally, we compare our model with PaDiM [6] and JNLD [34] using the same unified setting. The results on MVTec-AD dataset are presented in Table 3.

4.2. Qualitative Examples

Figure 5 illustrates the performance of our AnomalyGPT in unsupervised anomaly detection, and Figure 6 showcases the results in the 1-shot in-context learning. Our model is capable of indicating the presence of anomalies, pinpointing their locations, and providing pixel-level localization results. Users can engage in multi-turn dialogues related to

image content. In the 1-shot in-context learning setting, due to the absence of training, the model's localization performance is slightly lower than the unsupervised setting. More qualitative examples can be found in Appendix D.

4.3. Ablation Studies

To prove the efficacy of each proposed module, extensive ablation experiments are conducted on both the MVTec-AD and VisA datasets. We primarily focus on four aspects: the decoder, prompt learner, the usage of LLM for inference, and the utilization of LoRA to fine-tune the LLM. The principal results are presented in Table 4. Unsupervised training and testing are carried out on the MVTec-AD dataset, while the one-shot performance is evaluated on the visa dataset. It can be observed that the decoder demonstrates impressive pixel-level anomaly localization performance. Compared to manually-set thresholds, the LLM exhibits superior inference accuracy and provides additional functionality. Furthermore, prompt tuning outperforms LoRA in terms of accuracy and transferability.

5. Conclusion

We introduce AnomalyGPT, a novel conversational IAD vision-language model, leveraging the powerful capabilities of LVLM. AnomalyGPT can determine whether an image contains anomalies and pinpoint their locations without the need for manually specified thresholds. Furthermore, AnomalyGPT enables multi-turn dialogues focused on anomaly detection and demonstrates remarkable performance in few-shot in-context learning. The effectiveness of AnomalyGPT is validated on two common datasets. Our work delves into the potential application of large visual language models in anomaly detection, offering fresh ideas and possibilities for the field of industrial anomaly detection.

References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9592–9600, 2019.

[2] Xuhai Chen, Yue Han, and Jiangning Zhang. A zero-/few-shot anomaly classification and segmentation method for cvpr 2023 vand workshop challenge tracks 1&2: 1st place on zero-shot ad and 4th place on few-shot ad. *arXiv preprint arXiv:2305.17382*, 2023.

[3] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

[4] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

[5] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *arXiv preprint arXiv:2005.02357*, 2020.

[6] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *International Conference on Pattern Recognition*, pages 475–489. Springer, 2021.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190, 2023.

[9] Denis Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 98–107, 2022.

[10] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *European Conference on Computer Vision*, pages 303–319. Springer, 2022.

[11] Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. Winclip: Zero-/few-shot anomaly classification and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19606–19616, 2023.

[12] Sungwook Lee, Seunghyun Lee, and Byung Cheol Song. Cfa: Coupled-hypersphere-based feature adaptation for target-oriented anomaly localization. *IEEE Access*, 10: 78446–78454, 2022.

[13] Jiarui Lei, Xiaobo Hu, Yue Wang, and Dong Liu. Pyramidflow: High-resolution defect contrastive localization using pyramid normalizing flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14143–14152, 2023.

[14] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9664–9674, 2021.

[15] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[16] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[17] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[18] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. Ieee, 2016.

[19] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

[20] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *ACM SIGGRAPH 2003 Papers*, pages 313–318. 2003.

[21] Jonathan Pirnay and Keng Chai. Inpainting transformer for anomaly detection. In *International Conference on Image Analysis and Processing*, pages 394–406. Springer, 2022.

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[23] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14318–14328, 2022.

[24] Hannah M Schlüter, Jeremy Tan, Benjamin Hou, and Bernhard Kainz. Natural synthetic anomalies for self-supervised anomaly detection and localization. In *European Conference on Computer Vision*, pages 474–489. Springer, 2022.

[25] Yixuan Su, Tian Lan, Huayang Li, Jialu Xu, Yan Wang, and Deng Cai. Pandagpt: One model to instruction-follow them all. *arXiv preprint arXiv:2305.16355*, 2023.

[26] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[27] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *arXiv preprint arXiv:2305.11175*, 2023.

[28] Julian Wyatt, Adam Leach, Sebastian M Schmon, and Chris G Willcocks. Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 650–656, 2022.

[29] Guoyang Xie, Jingbao Wang, Jiaqi Liu, Feng Zheng, and Yaochu Jin. Pushing the limits of fewshot anomaly detection in industry vision: Graphcore. *arXiv preprint arXiv:2301.12082*, 2023.

[30] Xudong Yan, Huaidong Zhang, Xuemiao Xu, Xiaowei Hu, and Pheng-Ann Heng. Learning semantic context from normal samples for unsupervised anomaly detection. In *Proceedings of the AAAI conference on artificial intelligence*, pages 3110–3118, 2021.

[31] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian conference on computer vision*, 2020.

[32] Zhiyuan You, Lei Cui, Yujun Shen, Kai Yang, Xin Lu, Yu Zheng, and Xinyi Le. A unified model for multi-class anomaly detection. *Advances in Neural Information Processing Systems*, 35:4571–4584, 2022.

[33] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, 2021.

[34] Ying Zhao. Just noticeable learning for unsupervised anomaly localization and detection. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 01–06. IEEE, 2022.

[35] Ying Zhao. Omnial: A unified cnn framework for unsupervised anomaly localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3924–3933, 2023.

[36] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[37] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pretraining for anomaly detection and segmentation. In *European Conference on Computer Vision*, pages 392–408. Springer, 2022.

Supplementary Material

AnomalyGPT: Detecting Industrial Anomalies using Large Vision-Language Models

A. More Experimental Results of Existing IAD methods

As described in the paper, existing IAD methods solely provide anomaly scores for test samples. However, anomaly scores alone do not allow users to determine the presence of anomalies because they don't know the threshold to distinguish between normal and abnormal samples. The threshold varies considerably for each category of objects. Only by concurrently obtaining both normal and anomalous samples for each object category, along with their respective anomaly scores, can users identify the optimal classification threshold, which contradicts the original intent of IAD task that only utilize normal data.

One potential solution to this problem involves utilizing the maximum anomaly score from the normal samples of each category in the training set as the threshold. However, this approach is solely suited for "one-class-one-model" methods, which are designed specifically for detecting objects of a certain category within a specific environment. When presented with a previously unseen test sample, the model remains uncertain about which threshold to apply for decision-making. Thus, a unified threshold applicable across all categories is needed, which is challenging for existing IAD techniques.

We conduct experiments on two representative IAD methods, PatchCore [23] and WinCLIP [11]. PatchCore [23] achieves an Image-level AUC of 99.3% on the MVTec-AD dataset, while WinCLIP [11] is the state-of-the-art method for few-shot IAD. We assess the accuracy of both methods across individual categories at varying thresholds. It can be observed that the threshold exerts a significant influence on the performance of these two methods. Furthermore, a singular threshold displays markedly different efficacies across disparate categories. Hence, it becomes challenging to ascertain an optimal threshold unless experimental trials are conducted on test sets containing anomalous samples for each category. Figures 7 and Figure 8 delineate the outcomes of PatchCore [23] and WinCLIP [11] on the MVTec-AD [1] dataset across each category under different threshold settings.

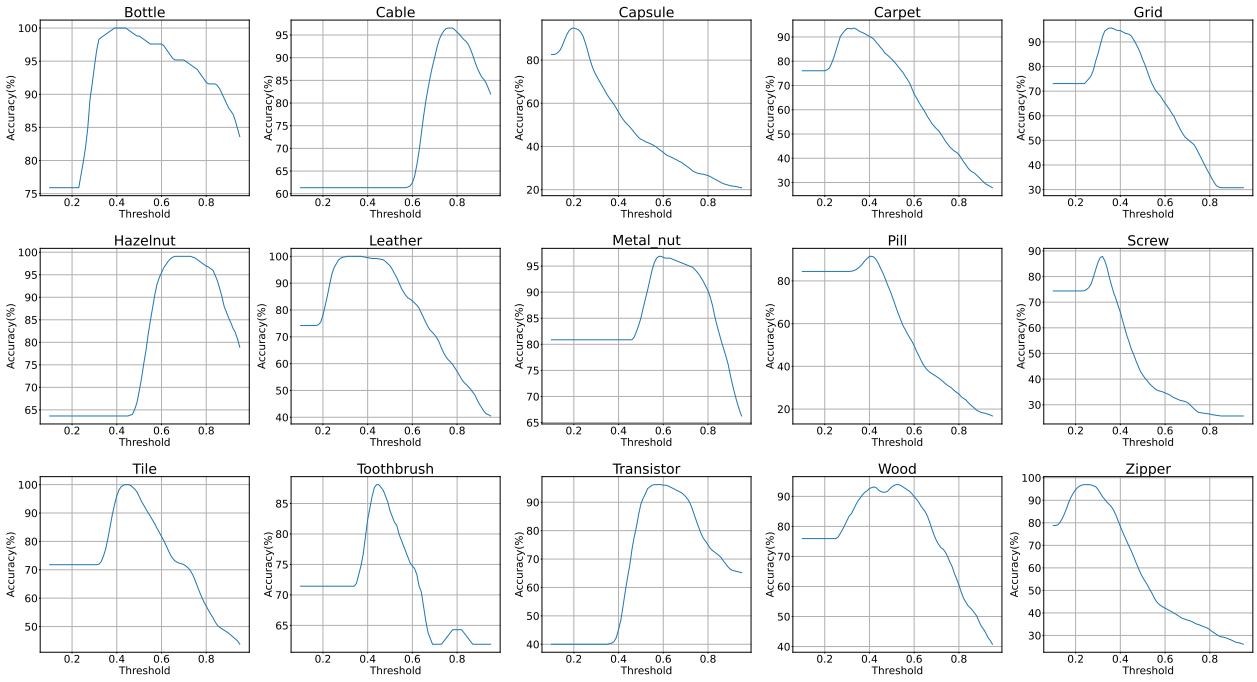

Figure 7. Experimental results of PatchCore [23] on the MVTec-AD [1] dataset across each category under different thresholds. The optimal threshold varies considerably for each category of objects.

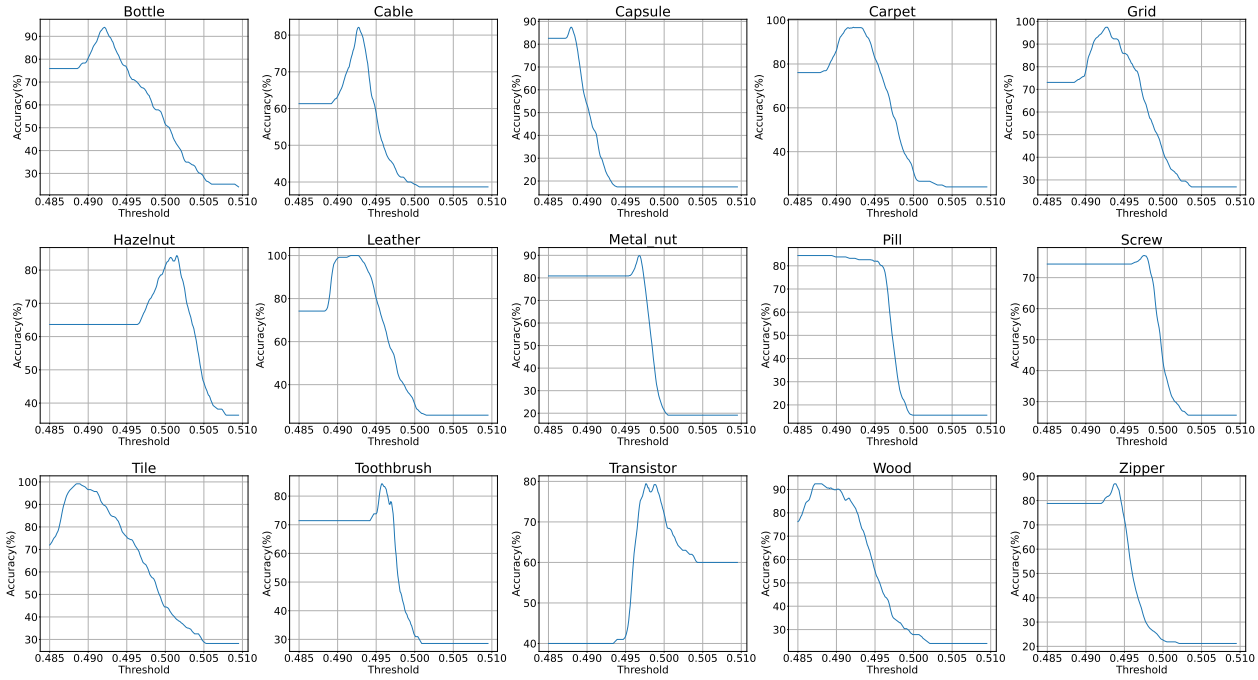

Figure 8. Experimental results of WinCLIP [11] on the MVTec-AD [1] dataset across each category under different thresholds. The optimal threshold varies considerably for each category of objects.

B. Normal and Abnormal Texts

Following WinCLIP [11], we utilize the compositional prompt ensemble to obtain texts presenting normality and abnormality. Specifically, we consider two levels of texts: (a) state-level, and (b) template level. The complete text can be composed by replacing the token `[c]` in a template-level text with one of state-level text and replacing the token `[o]` with the object's name. When the item's name is unavailable, the term "object" is adopted as the name for the item. Table 5 provides a detailed list of the multi-level texts.

(a) *State*-level (normal)

- `c := "[o]"`
- `c := "flawless [o]"`
- `c := "perfect [o]"`
- `c := "unblemished [o]"`
- `c := "[o] without flaw"`
- `c := "[o] without defect"`
- `c := "[o] without damage"`

State-level (anomaly)

- `c := "damaged [o]"`
- `c := "broken [o]"`
- `c := "[o] with flaw"`
- `c := "[o] with defect"`
- `c := "[o] with damage"`

(b) *Template*-level

- `"a cropped photo of the [c]."`
- `"a cropped photo of a [c]."`
- `"a close-up photo of a [c]."`
- `"a close-up photo of the [c]."`
- `"a bright photo of a [c]."`
- `"a bright photo of the [c]."`
- `"a dark photo of the [c]."`
- `"a dark photo of a [c]."`
- `"a jpeg corrupted photo of a [c]."`
- `"a jpeg corrupted photo of the [c]."`
- `"a blurry photo of the [c]."`
- `"a blurry photo of a [c]."`
- `"a photo of a [c]."`
- `"a photo of the [c]."`
- `"a photo of a small [c]."`
- `"a photo of the small [c]."`
- `"a photo of a large [c]."`
- `"a photo of the large [c]."`
- `"a photo of the [c] for visual inspection."`
- `"a photo of a [c] for visual inspection."`
- `"a photo of the [c] for anomaly detection."`
- `"a photo of a [c] for anomaly detection."`

Table 5. Lists of multi-level texts considered in this paper to present normal and abnormal semantics.

C. Detailed Image Description

As mentioned in the paper, prompts fed to the LLM typically follow the format:

Human: <Img> E_{img} </Img> E_{prompt} [Image Description] Is there any anomaly in the image? ### Assistant:

The *[Image Description]* part involves a description of the input image, providing information about the objects present in the image and their expected attributes. Such description furnishes the LVLM with foundational knowledge of the input image, aiding in the model's better comprehension of the image contents. The detailed description of every category in MVTec-AD [1] and VisA [37] datasets can be found in Table 6 and Table 7. Note that users can omit this descriptive input, and the model is still capable of performing IAD task based solely on the provided image input.

Class	Image description
Bottle	This is a photo of a bottle for anomaly detection, which should be round and without any damage, flaw, defect, scratch, hole or broken part.
Cable	This is a photo of three cables for anomaly detection, they are green, blue and grey, which cannot be missed or swapped and should be without any damage, flaw, defect, scratch, hole or broken part.
Capsule	This is a photo of a capsule for anomaly detection, which should be black and orange, with print '500' and without any damage, flaw, defect, scratch, hole or broken part.
Carpet	This is a photo of carpet for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Grid	This is a photo of grid for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Hazelnut	This is a photo of a hazelnut for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Leather	This is a photo of leather for anomaly detection, which should be brown with patterns and without any damage, flaw, defect, scratch, hole or broken part.
Metal nut	This is a photo of a metal nut for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part, and shouldn't be fliped.
Pill	This is a photo of a pill for anomaly detection, which should be white, with print 'FF' and red patterns and without any damage, flaw, defect, scratch, hole or broken part.
Screw	This is a photo of a screw for anomaly detection, whose tail should be sharp, and without any damage, flaw, defect, scratch, hole or broken part.
Tile	This is a photo of tile for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Toothbrush	This is a photo of a toothbrush for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Transistor	This is a photo of a transistor for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Wood	This is a photo of wood for anomaly detection, which should be brown with patterns and without any damage, flaw, defect, scratch, hole or broken part.
Zipper	This is a photo of a zipper for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.

Table 6. Detailed image description for every category in MVTec-AD dataset. The description will be added to the prompts of the corresponding category during training to provide foundational knowledge of the input image.

Class	Image description
Candle	This is a photo of 4 candles for anomaly detection, every candle should be round, without any damage, flaw, defect, scratch, hole or broken part.
Capsules	This is a photo of many small capsules for anomaly detection, every capsule is green and should be without any damage, flaw, defect, scratch, hole or broken part.
Cashew	This is a photo of a cashew for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Chewinggum	This is a photo of a chewinggum for anomaly detection, which should be white, without any damage, flaw, defect, scratch, hole or broken part.
Fryum	This is a photo of a fryum for anomaly detection on green background, which should be without any damage, flaw, defect, scratch, hole or broken part.
Macaroni1	This is a photo of 4 macaronis for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Macaroni2	This is a photo of 4 macaronis for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
PCB1	This is a photo of PCB for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
PCB2	This is a photo of PCB for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
PCB3	This is a photo of PCB for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
PCB4	This is a photo of PCB for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.
Pipe fryum	This is a photo of a pipe fryum for anomaly detection, which should be without any damage, flaw, defect, scratch, hole or broken part.

Table 7. Detailed image description for every category in VisA dataset. The description will be added to the prompts of the corresponding category during training to provide foundational knowledge of the input image.

D. More Qualitative Examples

We compare our approach with several existing LVLMs, specifically selecting PandaGPT [25], MiniGPT-4 [36], and LLaVA [17] for comparative analysis. We conduct experiments across various categories of both normal and anomalous samples. The results are presented in Figure 9, Figure 10, Figure 11, Figure 12, Figure 13 and Figure 14. It can be observed that only our method exhibits proficiency in both accurately answering questions related to anomaly detection and those about image content. In contrast, the other models demonstrate suboptimal performance in discerning the presence of anomalies and pinpointing their precise locations. Notably, PandaGPT and LLaVA show a marked tendency to misclassify all samples as anomalous. Conversely, MiniGPT-4 tends to err on the side of caution, predominantly labeling samples as normal.

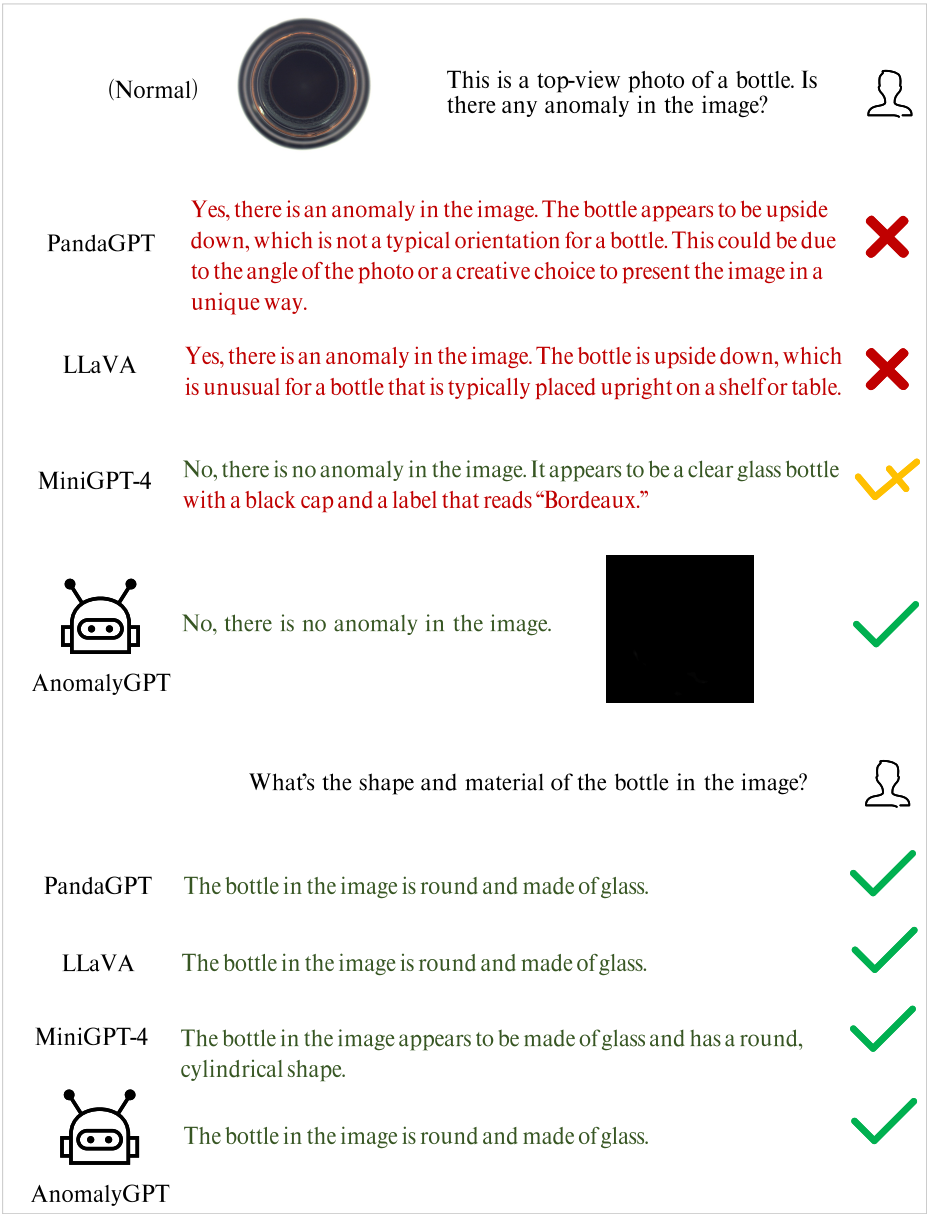

Figure 9. Comparison between AnomalyGPT, PandaGPT, LLaVA and MiniGPT-4. The input image is a top-view photo of a normal bottle. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

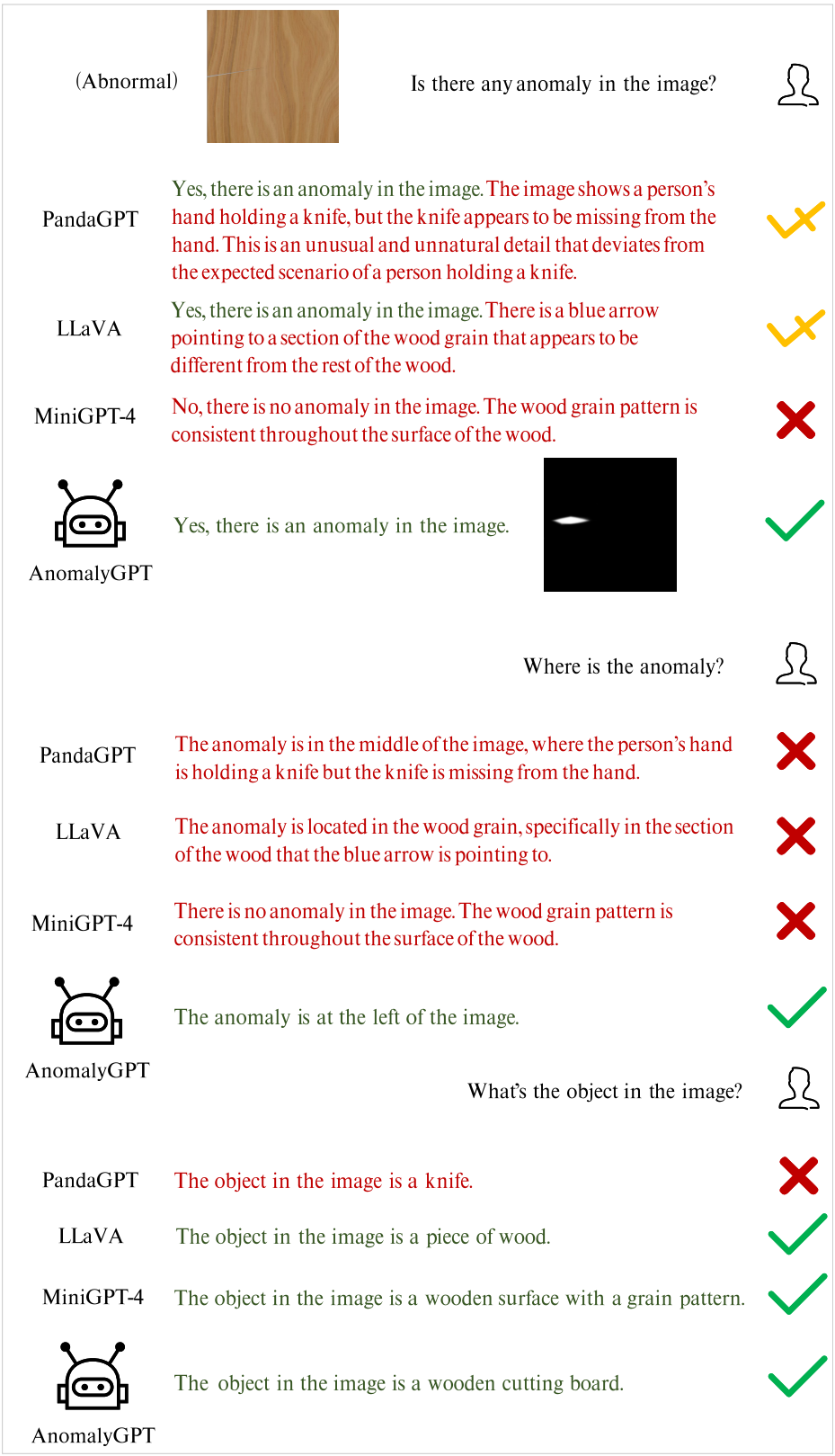

Figure 10. Comparison between AnomalyGPT, PandaGPT, LLaVA and MiniGPT-4. The input image is a photo of a cutting wood. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

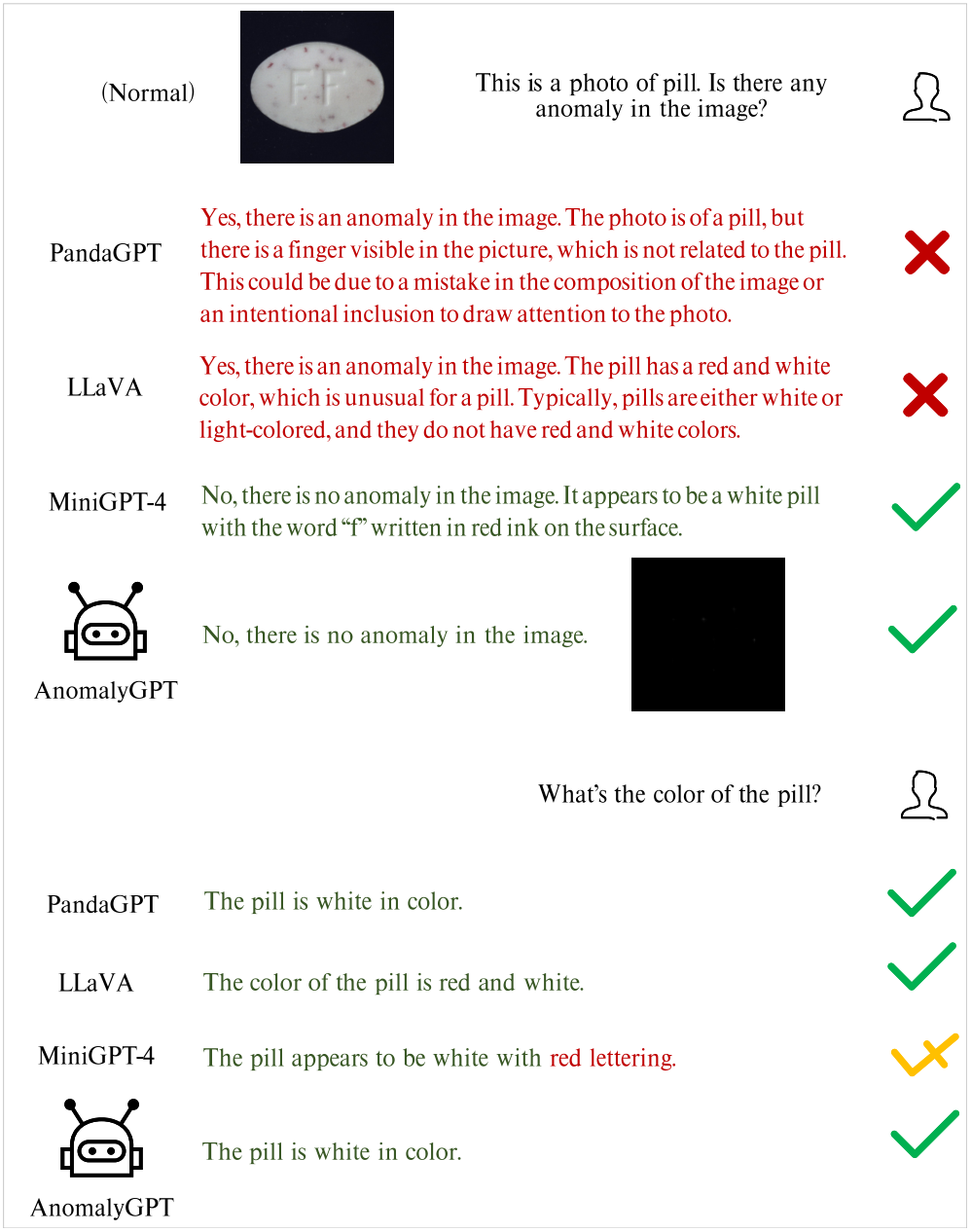

Figure 11. Comparison between AnomalyGPT, PandaGPT, LLaVA and MiniGPT-4. The input image is a photo of a normal pill. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

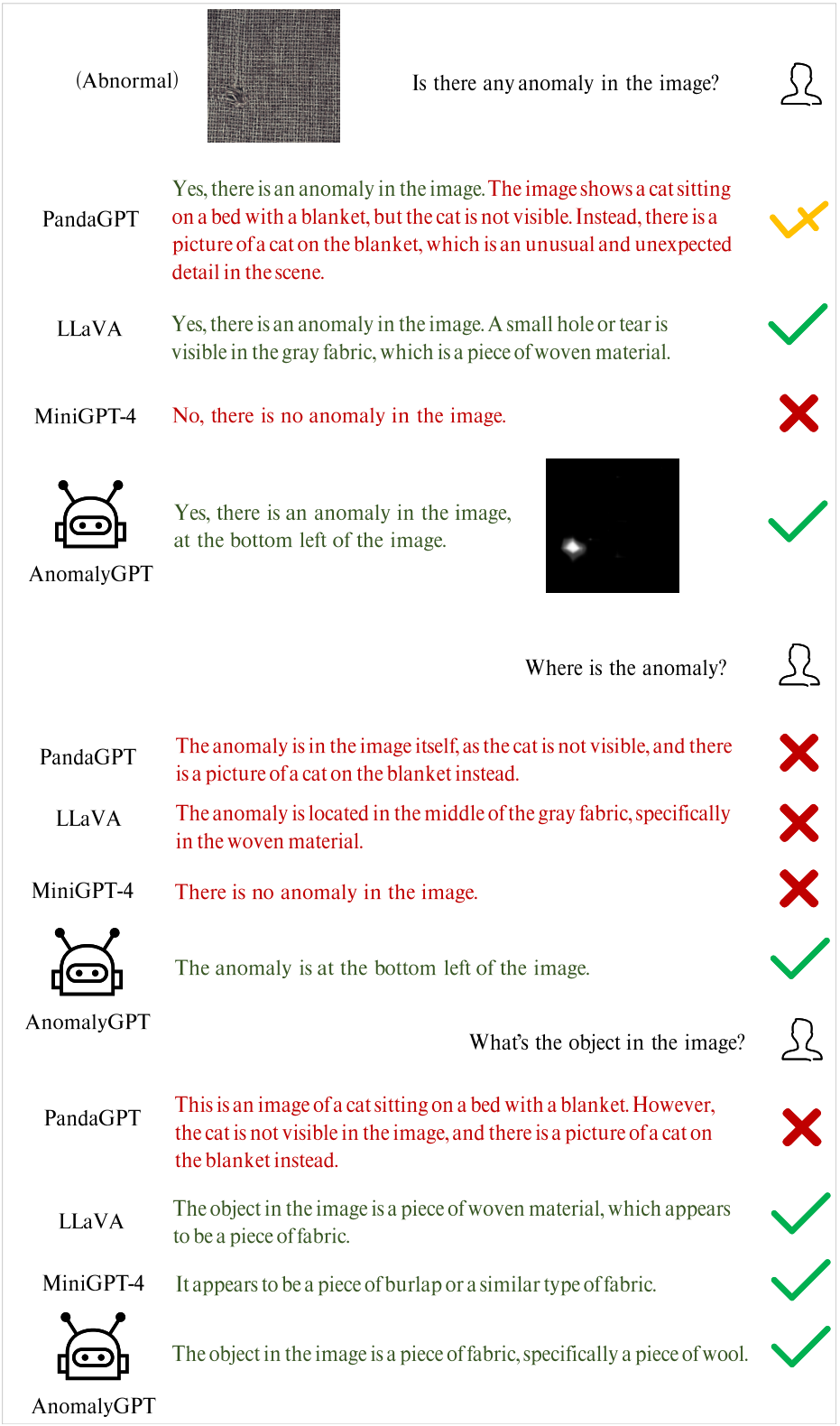

Figure 12. Comparison between AnomalyGPT, PandaGPT, LLaVA and MiniGPT-4. The input image is a photo of a piece of farbic with hole. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

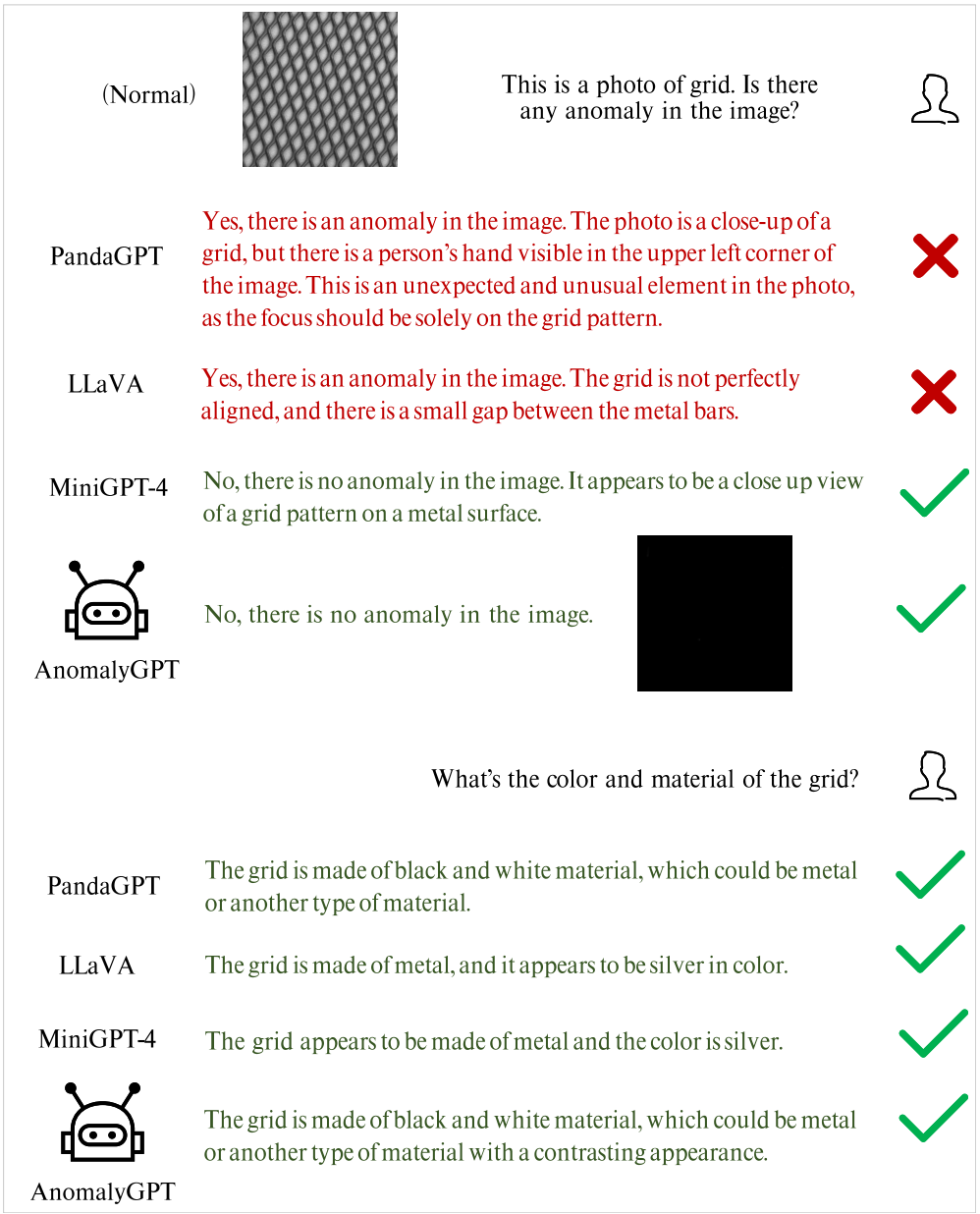

Figure 13. Comparison between AnomalyGPT, PandaGPT, LLaVA and MiniGPT-4. The input image is a photo of normal metal grid. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

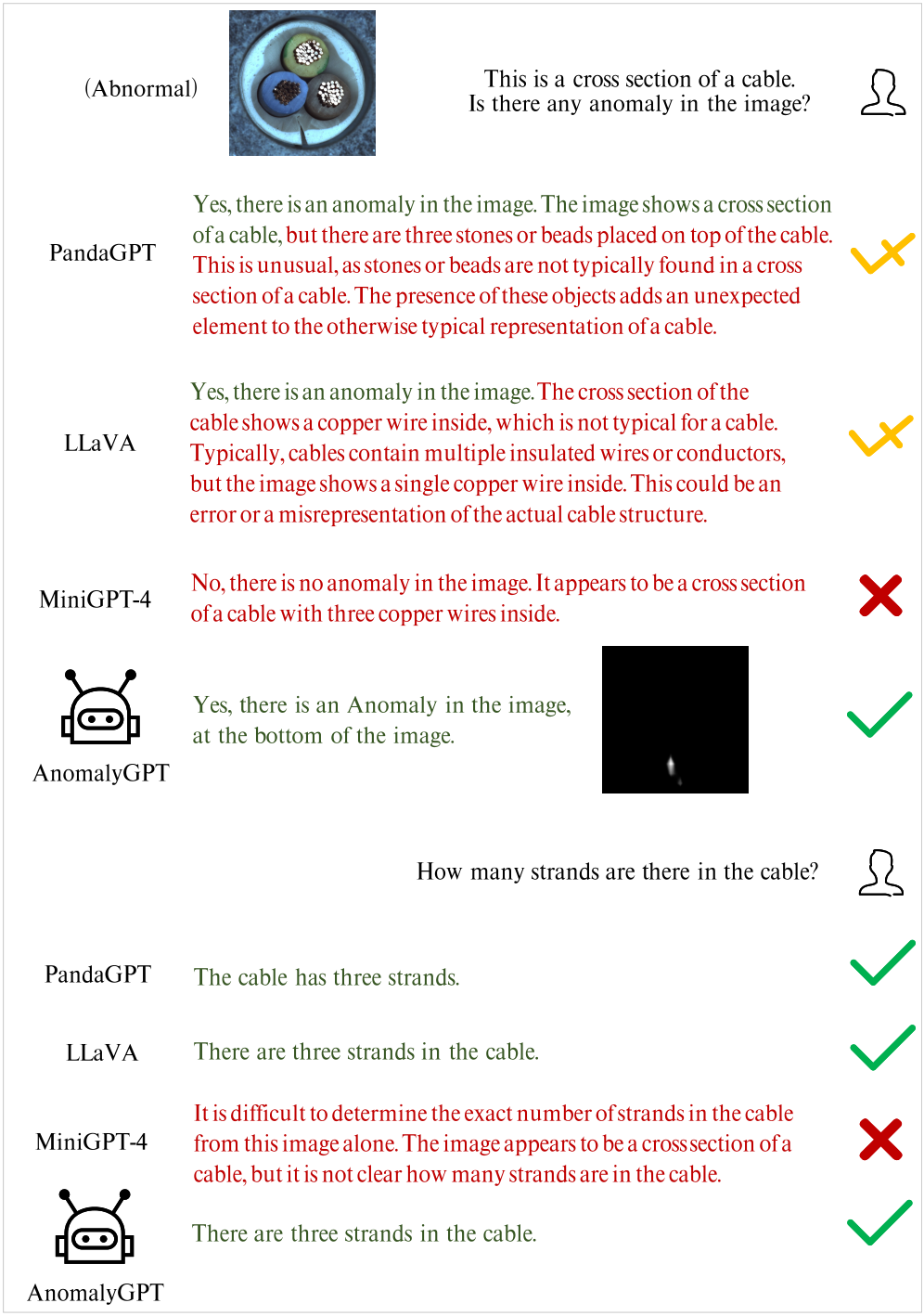

Figure 14. Comparison between AnomalyGPT, PandaGPT, LLaVA and MiniGPT-4. The input image is a photo of a cable with defect. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.