南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

AnomalyGPT: Detecting Industrial Anomalies Using Large Vision-Language Models

Zhaopeng Gu[1,2*], Bingke Zhu[1,3*], Guibo Zhu[1,2†], Yingying Chen[1,3†], Ming Tang[1,2], Jinqiao Wang[1,2,3]

[1]Foundation Model Research Center, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3]Objecteye Inc., Beijing, China
guzhaopeng2023@ia.ac.cn, {bingke.zhu, gbzhu, yingying.chen, tangm, jqwang}@nlpr.ia.ac.cn

AAAI 2024

Background

IAD (Industrial Anomaly Detection) task: aims to detect and localize anomalies in industrial 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.

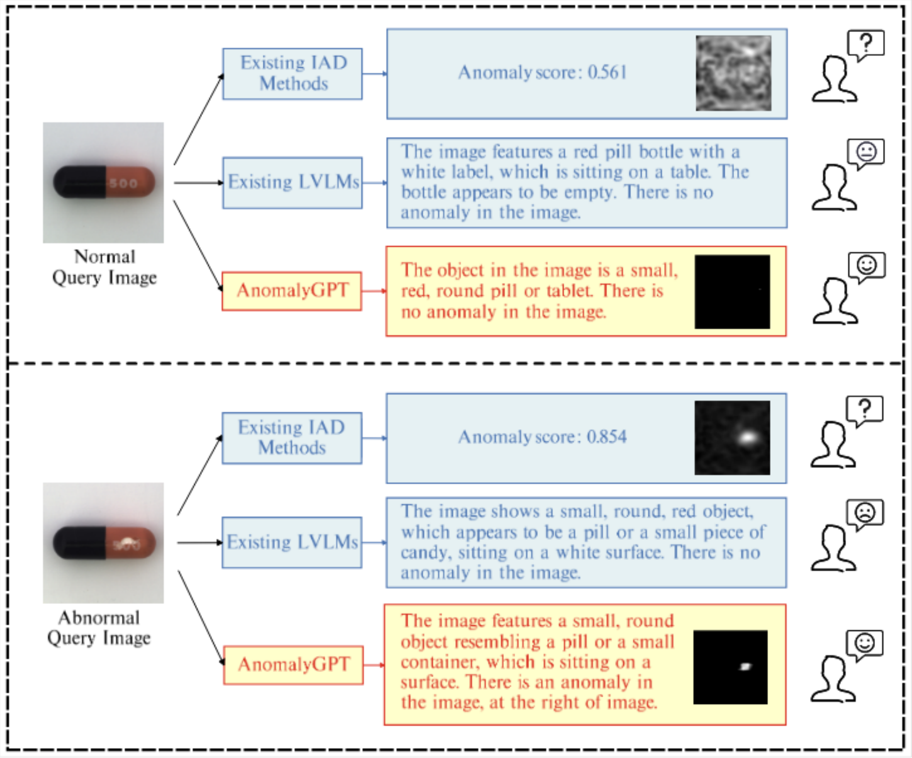


Figure 1: Comparison between our AnomalyGPT, existing IAD methods and existing LVLMs. Existing IAD methods can only provide anomaly scores and need manually threshold setting, while existing LVLMs cannot detect anomalies in the image. AnomalyGPT can not only provide information about the image but also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

Related works

Existing IAD methods:

Reconstruction-based aim to reconstruct anomalous samples to their corresponding normal counterparts and detect anomalies by calculating the reconstruction error (network architectures rang from autoencoder and GAN to Transformer and diffusion model).

Feature embedding-based modeling the feature embeddings of normal samples.

1. Approaches such as PatchSVDD aim to find a hypersphere that tightly encapsulates normal samples.
2. PyramidFlow use normalizing flows to project normal samples onto a Gaussian distribution.
3. CFA 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.

one-class-one-model: impractical for novel object categories and less suitable for dynamic production environments.

Motivation

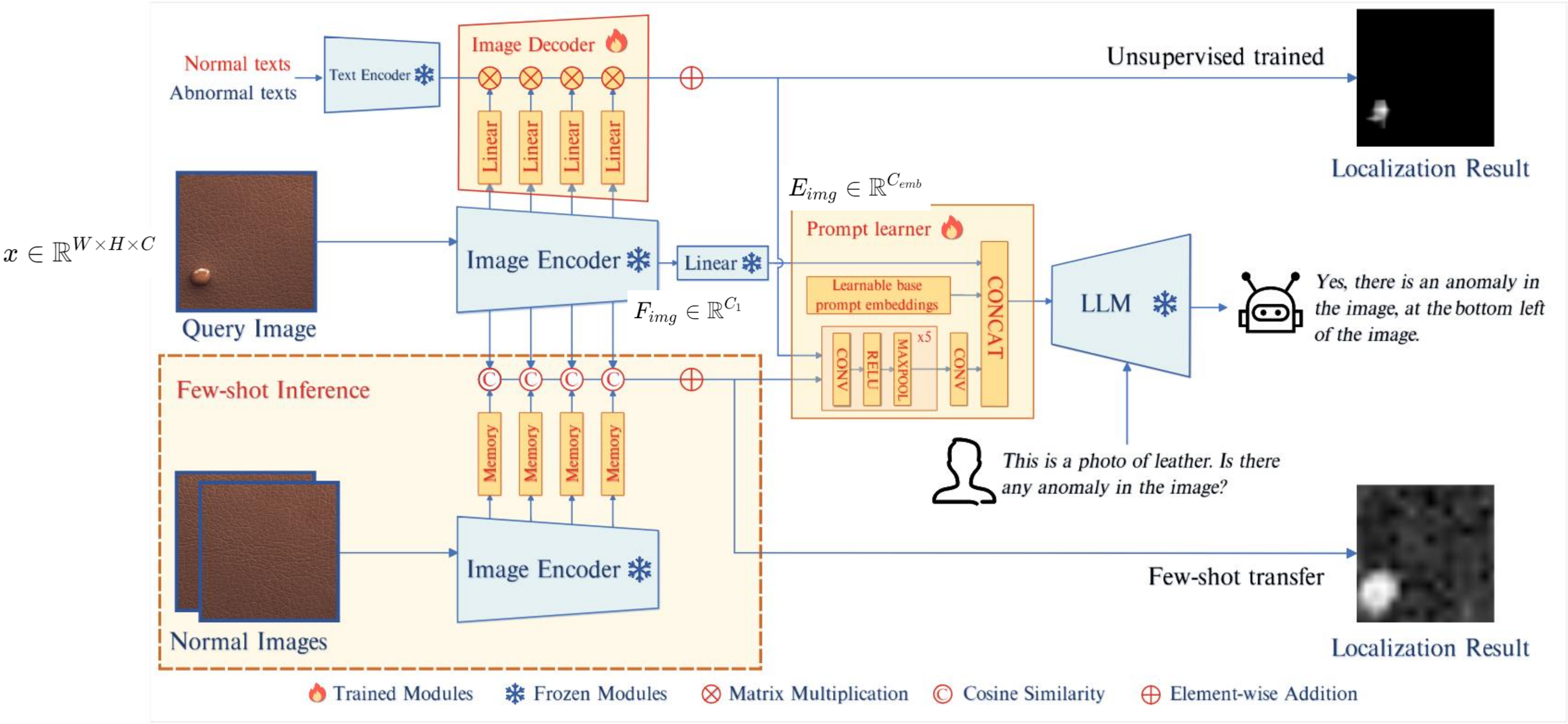
Data scarcity:
Methods like LLaVA and PandaGPT are pre-trained on 160k images with corresponding multi-turn dialogues. IAD datasets contain only a few thousand samples, rendering direct fine-tuning easy to overfitting and catastrophic forgetting.

Solution: using prompt embeddings to fine-tune the LVLM instead of parameter fine-tuning.

Fine-grained semantic:
The limitation of the LLM’s weaker discernment of fine-grained semantic.

Solution: We propose a lightweight, visual-textual feature-matching-based decoder to generate pixel-level anomaly localization results.

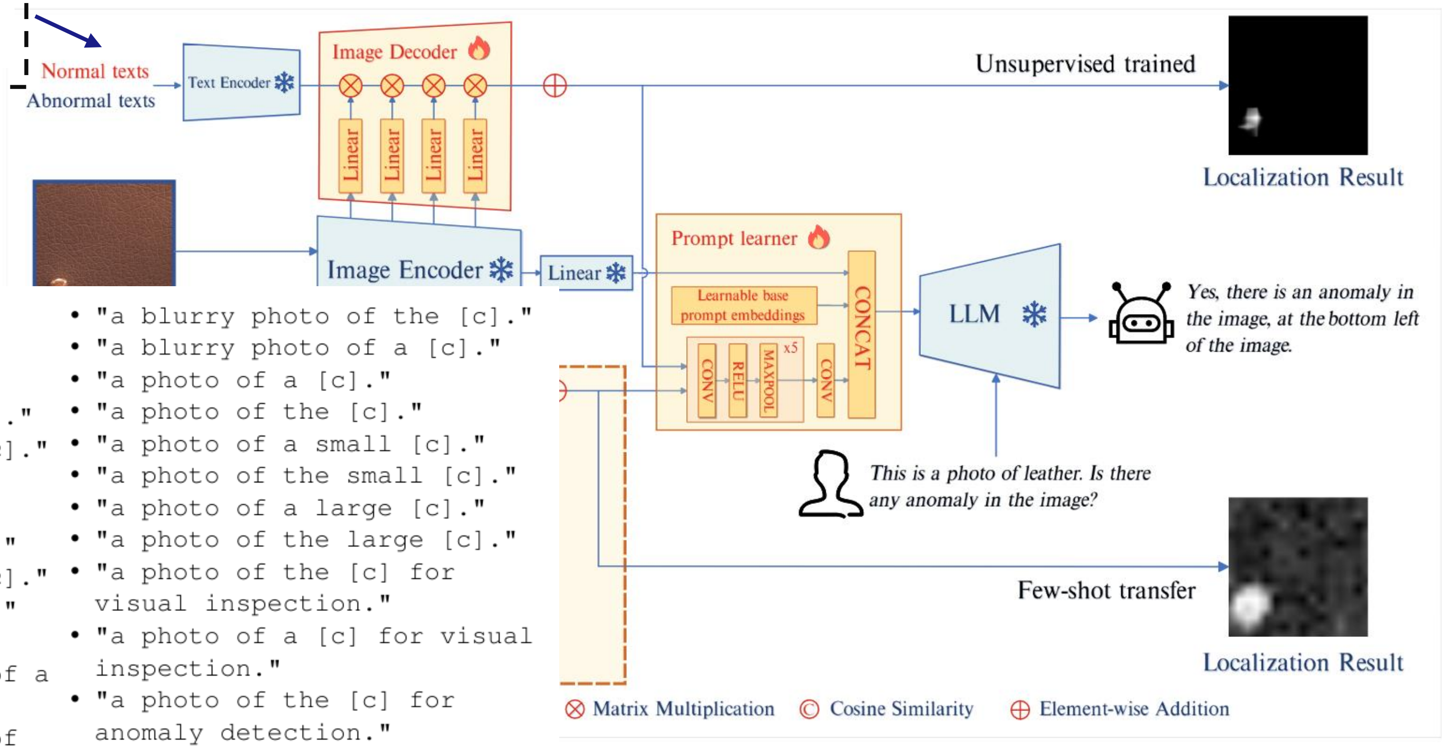
Method/Decoder and Prompt Learner



Method/Decoder and Prompt Learner

A photo of a normal bottle.
A photo of an abnormal capsule.

$$F_{text} \in \mathbb{R}^{2\times C_{text}}$$



(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."

Method/Decoder and Prompt Learner

A photo of a normal bottle.
A photo of an abnormal capsule. $F_{text} \in \mathbb{R}^{2\times C_{text}}$

$F^i_{patch} \in \mathbb{R}^{H_i\times W_i\times C_i}$

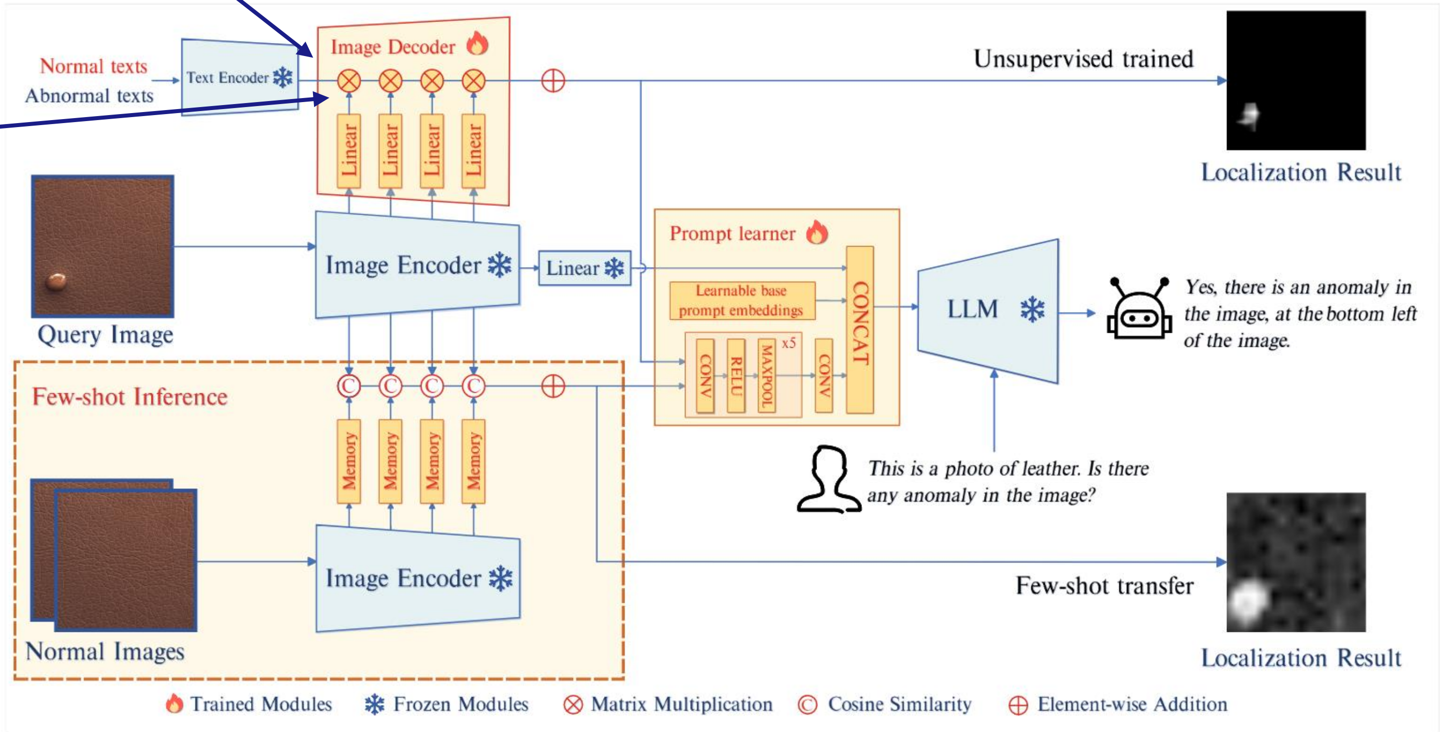
$\tilde{F}^i_{patch} \in \mathbb{R}^{H_i\times W_i\times C_{text}}$

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

$B^i \in \mathbb{R}^{N\times C_i}$

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

$M \in R^{H\times W}$



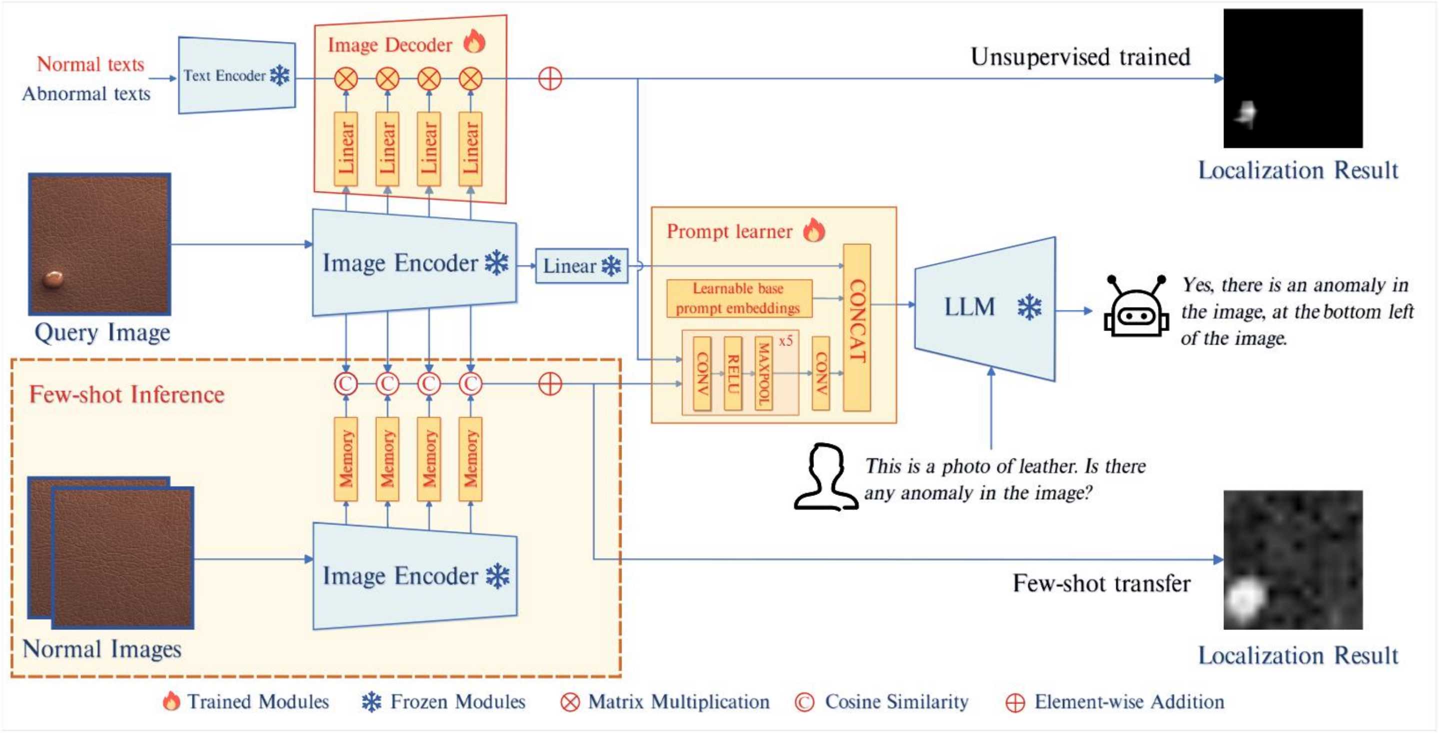
Method/Decoder and Prompt Learner

$$E_{base} \in \mathbb{R}^{n_1 \times C_{emb}}$$

$$M \in R^{H \times W} \longrightarrow E_{dec} \in \mathbb{R}^{n_2 \times C_{emb}}$$

$$\longrightarrow E_{prompt} \in \mathbb{R}^{(n_1+n_2) \times C_{emb}}$$

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.



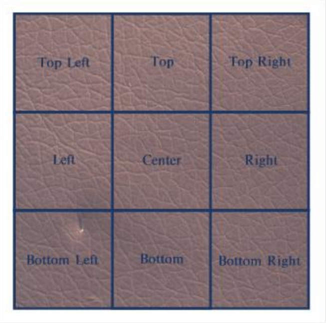
Method/Decoder and Prompt Learner

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

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

Yes, there is an anomaly in the image, at the bottom left of the image.
or *No, there are no anomalies in the image*.



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.

The disparity between the text sequence generated by the model and the target text sequence (where *n* is the number of tokens).

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

In IAD task, where most regions in anomaly images are still normal, employing focal loss can mitigate the problem of class imbalance (where $n = H \times W$ represents the total number of pixels, *yi* is the output of decoder and *yˆi* is the ground truth value).

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

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

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

Experiment

Datasets:

MVTec-AD comprises 3629 training images and 1725 testing images across 15 different categories.

VisA contains 9621 normal images and 1200 anomalous images across 12 categories.

Consistent with previous IAD methods, we only use the normal data from these datasets for training.

Anomaly Simulation

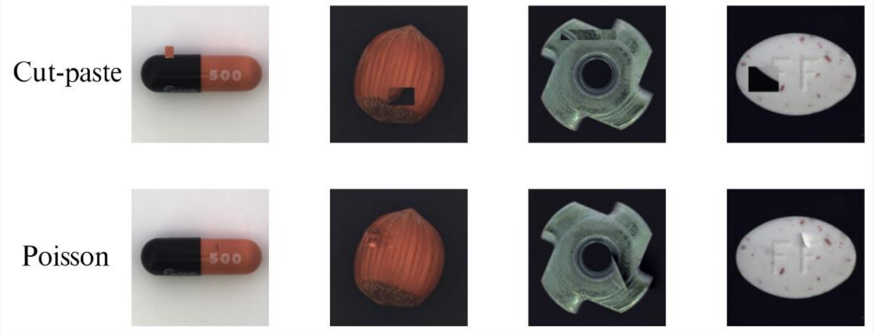


Figure 3: Illustration of the comparison between cut-paste and poisson image editing. The results of cut-paste exhibit evident discontinuities and the results of poisson image editing are more natural.

Experiment

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 (Jeong et al. 2023).

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 (Zhao 2023).

Experiment

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.

Experiment

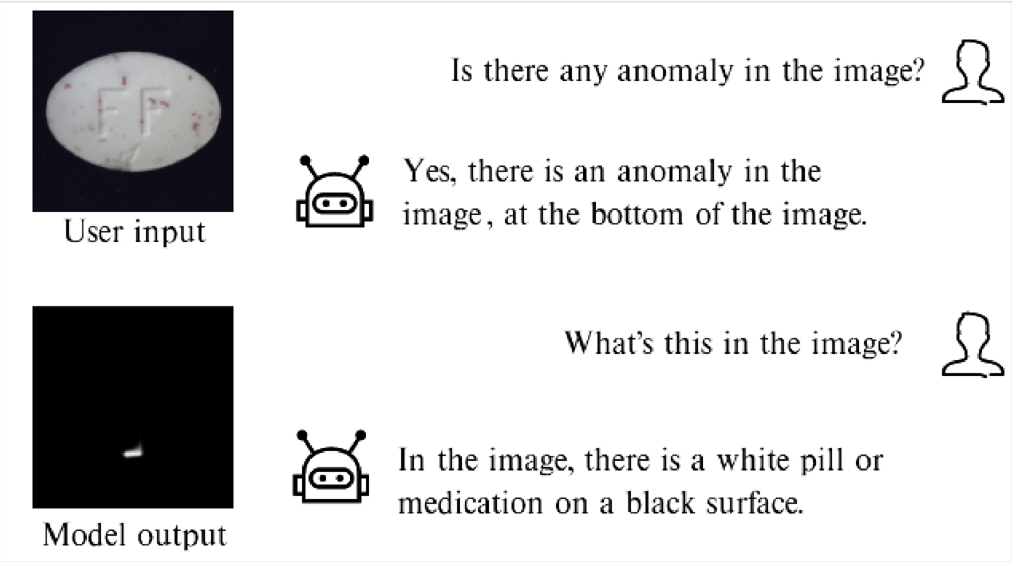


Figure 5: Qualitative example of AnomalyGPT in the unsupervised setting. AnomalyGPT is capable of detecting anomaly, pinpointing its location, providing pixel-level localization results and answering questions about the image.

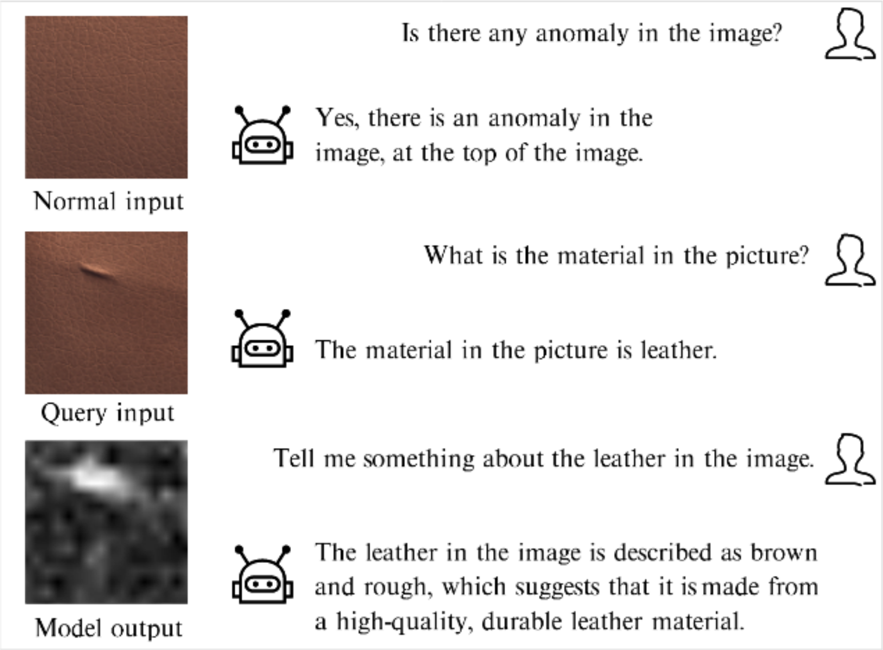


Figure 6: Qualitative example of AnomalyGPT in the one-normal-shot setting. The localization performance is slightly lower compared to the unsupervised setting due to the absence of parameter training.

Thank you