AR-Pro: Anomaly Explanation and Repair with Formal Properties

Xiayan Ji* **Anton Xue*** **Eric Wong** **Oleg Sokolsky** **Insup Lee**
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
{xjiae, antonxue, exwong, sokolsky, lee}@seas.upenn.edu

Abstract

Anomaly detection is crucial for identifying critical errors and suspicious behaviors. However, current methods struggle to explain why an input is anomalous in a rigorous manner. To address this explainability gap, we investigate counterfactual explanations supported by formal properties to ensure their quality. Given an anomalous input, our explanation is a "repair" highlighting what a non-anomalous analog should look like, removing the abnormality while preserving the normality and similarity to the input. We designed four properties that generalize across various domains, formulating a framework AR-Pro for explainable anomaly detection. Empirically, our framework produces semantically meaningful repairs that improve formal criteria by 96.14% on the VisA dataset for visual anomalies, and by 61.25% on the SWaT dataset for time-series anomalies.

1 Introduction

The principal objective of anomaly detection is to determine whether data deviates from an established norm. A positive determination suggests that the data merits closer inspection. For instance, unusual network traffic can indicate potential malicious attacks, necessitating a review of security logs [8, 15]. In the context of civil infrastructure, irregular sensor readings may signal architectural vulnerabilities that require structural inspections [23, 33, 35]. In the financial sector, atypical transactions can hint at potential fraud, prompting authorities to conduct forensic audits [3, 31]. Additionally, analyzing unexpected phenomena in scientific data can lead to new knowledge discoveries [11, 27].

After detecting an anomalous input, explaining the root cause becomes a natural follow-up. Although there is some research on anomaly explanation [46], this area remains under-explored. Notably, while explanation techniques based on reconstruction error and gradient saliency [30] are effective in identifying *where* the input region is anomalous, they often fall short in clarifying *why* it is anomalous. This limitation can significantly undermine usability; less experienced users might overly rely on the model without understanding its reasoning, and experts may lose trust in the model if they cannot understand why their decisions mismatch the model's predictions.

This gap in explainability motivates our exploration of counterfactual explanations for anomaly detection, with a specific focus on anomaly repairs [52]. Such repairs aim to address the what-if question: 'What would a non-anomalous instance look like?' — which can help users better understand *why* an input is anomalous by showing how it deviates from the norm. For instance, healthcare professionals can use counterfactual explanations to understand what changes in a patient's chest X-ray might lead to improved diagnostic outcomes [40]. In the industrial manufacturing sector, these explanations can clarify different types of product defects. While counterfactual explanations

*These authors contributed equally to this work.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

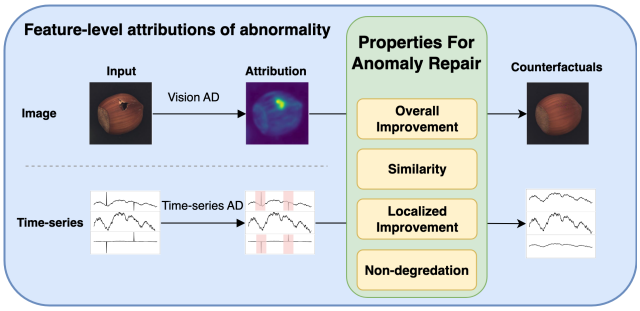


Figure 1: Overview of AR-Pro framework.

are widely recognized in explainable machine learning [38, 43], their application to anomaly detection is understudied.

However, merely generating a counterfactual explanation is not sufficient; we must also ensure its quality. An anomalous instance may have multiple possible repairs, but not all of them are of equal quality. Therefore, it is necessary to establish formal properties to systematically evaluate the quality of proposed repairs. For instance, a high-quality repair should not only improve the overall anomaly score but also closely resemble the original input, be localized to the region of error, and not degrade the normal areas. However, a challenge arises in that different domains (e.g., vision, time-series) often present unique considerations, complicating the establishment of universally applicable, mathematically formalized properties for explainability techniques [26].

In this work, we leverage the unique characteristics of anomaly detection methods to define the desired properties for counterfactual explanations. In particular, we use the fact that many existing anomaly detection methods can generate *feature-level attributions* of anomalies, which informs the definition of properties for effective counterfactual explanations. We couple this insight with recent advances in generative modeling to propose AR-Pro, a framework for guiding the generation of counterfactual repairs. The repairs generated under this framework demonstrate robust performance, respecting the specified properties across image and time-series anomaly datasets, particularly VisA and SWaT. Figure 1 provides an overview of our framework. In summary, our contributions are outlined as follows:

- We identify a set of semantically meaningful properties for anomaly repair applicable across different domains.
- Utilizing these properties, we develop a formal property-guided framework to generate high-quality repairs.
- Our experiments demonstrate that the guided framework produces semantically meaningful repairs that significantly outperform off-the-shelf diffusion models based on formal criteria.

The remainder of this paper is organized as follows: Section 2 provides an overview and formulates the formal properties for counterfactuals. Section 3 explains how these properties guide the generation of anomaly repairs. Section 4 presents empirical results on vision and time-series datasets. Section 6 discusses the limitations of our methods, concluding with Section 7.

2 Overview and Problem Formulation

As an overview, Section 2.1 reviews reconstruction-based anomaly detection and presents formalisms for feature attributions of abnormality. Building on these attributions, we introduce formal properties desirable for anomaly repairs in Section 2.2. Throughout this paper, the terms 'counterfactuals' and 'anomaly repair' will be used interchangeably.

2.1 Anomaly Detection and Attribution

The primary task of anomaly detection is to check whether some input data is *anomalous*. This is usually quantified by a scoring function $s : \mathbb{R}^n \to \mathbb{R}$ that computes an anomaly score $s(x)$ given some input $x \in \mathbb{R}^n$, where a larger anomaly score indicates a more anomalous input. In addition to the anomaly score, it is useful to localize the anomaly with an *anomaly map* $\alpha(x) = |x - \hat{x}|$, as

illustrated in Figure 2. By definition, $\alpha_i(x)$ has a high value if and only if the reconstruction for feature i is poor, necessitating further scrutiny for repair. Importantly, $\alpha(x)$ provides a feature-level measure of anomalies for x, although it is noteworthy that other valid definitions also exist.

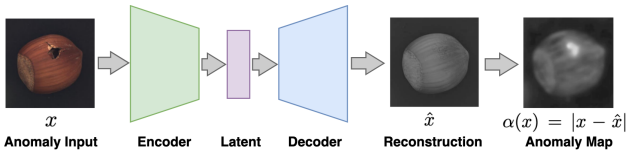


Figure 2: An overview of reconstruction-based anomaly detection; An anomalous image x is mapped to the reconstructed image $\hat{x}$, and the anomaly map $\alpha(x) = |x - \hat{x}| \in \mathbb{R}^n$ is normalized and plotted in gray scale. An ℓ^2-norm anomaly score is then $s(x) = |\alpha_1(x)|^2 + \cdots + |\alpha_n(x)|^2$.

In general, numerous relations can exist between the scoring function $s : \mathbb{R}^n \to \mathbb{R}$ and the anomaly map $\alpha : \mathbb{R}^n \to \mathbb{R}^n$. We present a linearly decomposable relation in Definition 2.1.

Definition 2.1. An attribution map α *linearly decomposes* the scoring function s with respect to $s_0 : \mathbb{R}^n \to \mathbb{R}$ and $s_1, \ldots, s_n : \mathbb{R} \to \mathbb{R}$ if for all $x \in \mathbb{R}^n$:

$$s(x) = s_0(x) + s_1(\alpha_1(x)) + \cdots + s_n(\alpha_n(x))$$

As illustrated in Figure 2, we have $s_0(x) = 0$ and each $s_i(\alpha_i(x)) = |\alpha_i(x)|^2$ for all $x \in \mathbb{R}^n$. While the decomposition outlined in Definition 2.1 may seem straightforward, this linear relationship between the feature attributions and the overall anomaly score is commonly encountered in practice. For instance, vision anomaly detectors [48], time series anomaly detectors [14, 41, 45], and text anomaly detectors [19] all derive their anomaly scores from a linear combination of feature attributions. Furthermore, even in cases where the anomaly detector does not rely on reconstruction, a linearly decomposing anomaly map can still be achieved using post-hoc feature attribution techniques such as gradient saliency [29].

Example 2.2. A maximum-likelihood estimation-based method for images is proposed in [48]. In this paradigm, the anomaly score is expressed as follows:

$$s(x) = \underbrace{\log|\det J(x)|}_{s_0(x)} + \sum_{i=1}^{n} \underbrace{\log p_i(x)}_{s_i(\alpha_i(x))}$$

where $J : \mathbb{R}^n \to \mathbb{R}^n$ and $p : \mathbb{R}^n \to \mathbb{R}^n$ are the change-of-variable Jacobian and likelihood estimations of the method. Crucially, the anomaly score is linearly decomposable.

2.2 Formal Properties for Anomaly Repair

Having presented the formalism of scoring functions and anomaly maps, we now concretize our formal properties for anomaly repair. Suppose that we are given some anomalous input $x_{\mathsf{bad}} \in \mathbb{R}^n$, the main objective of anomaly repair is to find some $x_{\mathsf{fix}} \in \mathbb{R}^n$ that *repairs* x_{bad}. Because anomalies are measured with respect to a given scoring function $s : \mathbb{R}^n \to \mathbb{R}$, the improvement in anomaly score can be expressed as follows:

$$s(x_{\mathsf{fix}}) \leq s(x_{\mathsf{bad}}) \qquad \text{(Property 1)}$$

However, the repair should not be arbitrary, and in particular, x_{fix} should closely resemble x_{bad}. This intuition is formed by the practical consideration that only a subset of features typically requires repair. We refer to this subset as the *anomalous region*. x_{fix} and x_{bad} may differ in the anomalous region but should be similar in the *normal region*. Therefore, to achieve similarity-preserving anomaly repair, one must first identify the anomalous region of x_{bad}.

A common approach for identifying the anomalous region involves thresholding the anomaly map $\alpha : \mathbb{R}^n \to \mathbb{R}^n$. Under this method, a feature x_i is considered anomalous if $\alpha_i(x) \geq \tau_i$ for a given threshold τ_i. We formalize this technique with a binary-valued *region selector* $\omega : \mathbb{R}^n \to \{0, 1\}^n$ as follows:

$$\omega(x) = \mathbb{I}[\alpha(x) \geq \tau], \qquad \tau \in \mathbb{R}^n$$

where $\mathbb{I}$ represents the vector-valued indicator function and τ refers to a set of thresholds that can be adjusted to tune the sensitivity of ω. Importantly, this setup facilitates a formal discussion of the similarity between the anomalous input and its repair, which we will present as follows:

$$\|(1 - \omega_{\mathsf{bad}}) \odot (x_{\mathsf{fix}} - x_{\mathsf{bad}})\| \approx 0 \qquad \text{(Property 2)}$$

where we use $\omega_{\mathsf{bad}} = \omega(x_{\mathsf{bad}})$ for brevity and $\odot$ denotes element-wise vector multiplication. Thus, the difference between x_{fix} and x_{bad} should be minimal in the normal region, as indicated by the binary vector $1 - \omega_{\mathsf{bad}}$. Although we typically employ a norm $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ to measure this difference, other similarity measures are also applicable.

However, Property 1 and Property 2 alone are insufficient to ensure that the anomalous region is effectively improved in x_{fix}. For example, x_{fix} might achieve a lower overall anomaly score than x_{bad}, yet display higher values on the feature-level anomaly map within the anomalous region. To address this, we explicitly focus on ensuring *improvement within a specific region*. Consequently, we introduce a notion to restrict the anomaly score to particular features. For $\beta \in \{0, 1\}^n$, we defined $s_\beta : \mathbb{R}^n \to \mathbb{R}$ as follows:

$$s_\beta(x) = s_0(x) + \sum_{i:\beta_i=1} s_i(\alpha_i(x)).$$

s_β represents the anomaly score s restricted by β, the binary vector indicating selected features. This adjustment enables us to formally establish a condition for score improvement in the anomalous region as follows:

$$s_{\omega_{\mathsf{bad}}}(x_{\mathsf{fix}}) \leq s_{\omega_{\mathsf{bad}}}(x_{\mathsf{bad}}) \qquad \text{(Property 3)}$$

The property stipulates that the score over ω_{bad} for x_{fix} should be lower than that for x_{bad}, indicating that x_{fix} has achieved local improvement by reducing its abnormality.

Moreover, it is crucial to ensure that x_{fix} does not significantly degrade in the normal region. While Property 1, Property 2, and Property 3 might be satisfied, it is possible that the proposed repair could inadvertently increase the anomaly score within the normal region. To prevent this, we explicitly define the following property to ensure that such a scenario does *not* occur:

$$\left[s_{1-\omega_{\mathsf{bad}}}(x_{\mathsf{fix}}) - s_{1-\omega_{\mathsf{bad}}}(x_{\mathsf{bad}})\right]_+ \approx 0 \qquad \text{(Property 4)}$$

where $[\,\cdot\,]_+$ is shorthand for $\max(0, \cdot)$.

In sum, Property 1 indicates an overall improvement in the anomaly score of the repair. Property 2 ensures the similarity to the input. Property 3 checks for the localized improvement of the anomaly score over the anomalous region, and Property 4 assesses the non-degradation in the normal region.

3 Formal Property-guided Anomaly Repair

In this section, we introduce the AR-Pro framework for anomaly repair, guided by formally defined properties. Typically, anomaly detectors generate anomaly scores s, and a linearly decomposing anomaly map α can be derived from the detector (reconstruction-based) or via a post-hoc feature attribution method. Therefore, we assume that the scoring function s, the anomaly map α, and the region selector ω are predefined.

Assumption 3.1. The scoring function s, the anomaly map α, and the region selector ω are given.

Given some anomalous input x_{bad}, our formalism of the repair model, $\mathcal{R}$, functions as follows:

$$x_{\mathsf{fix}} = \mathcal{R}(x_{\mathsf{bad}}, \alpha(x_{\mathsf{bad}}), \omega(x_{\mathsf{bad}}))$$

where x_{fix} is the proposed repair. At a high level, our repair process consists of two main components: a generative model and formal property guidance integrated into its generation process. The generative model is trained on non-anomalous data to learn its distribution and to produce outputs that resemble this non-anomalous distribution when given an anomalous input. In Section 3.1, we discuss the training of a diffusion-based model [10], which has demonstrated superior performance compared to other generative models. Subsequently, Section 3.2 formalizes how the desired repair properties outlined in Section 2.2 are translated into loss functions. Finally, Section 3.3 details how formal property guidance is applied during the generation process of the anomaly repair.

3.1 Training of Generative Models

Given the rarity of anomalies, we train the generative models exclusively on non-anomalous inputs to generate outputs that closely resemble their distribution. Given the efficacy of diffusion models in producing high-quality results across various domains, such as images [10], language [16], and time-series [49], we select them as our primary generative models. It is important to note that other generative models like Variational Autoencoders (VAE) [13], Generative adversarial Networks (GAN) [9], and Transformers [42] could also be utilized, following similar training principles. We employ a standard weighted mean squared error loss for training the approximator model $\mu_\theta(x_t, t)$ [10]: $\mathcal{L}_\theta = \sum_{t=1}^{T} \mathbb{E}_{q(x_t|x_0)} \left[\|\mu(x_t, x_0) - \mu_\theta(x_t, t)\|^2\right]$ for T diffusion steps.

3.2 From Properties to Objective Functions

We now express our formal properties as objectives that guide the iterative generation process of the repair model. Specifically, we define the loss function in terms of four distinct components, as follows:

$$\mathcal{L}(x_{\mathsf{bad}}, x_{\mathsf{fix}}) = \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 + \lambda_3\mathcal{L}_3 + \lambda_4\mathcal{L}_4$$

where $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3, \mathcal{L}_4$ denote the loss functions associated with each of the four properties, and $\lambda_1, \lambda_2, \lambda_3, \lambda_4 > 0$ are hyperparameters. The primary objective of anomaly repair is to reduce the anomaly score. Consequently, the loss corresponding to Property 1 is defined as follows:

$$\mathcal{L}_1 = s(x_{\mathsf{fix}})$$

Next, we address the Property 2 loss. This component focuses on ensuring that x_{fix} and x_{bad} are similar in the normal region. We define this requirement as follows:

$$\mathcal{L}_2 = \|(1 - \omega_{\mathsf{bad}}) \odot (x_{\mathsf{fix}} - x_{\mathsf{bad}})\|_2^2$$

For Property 3, we introduce a penalty when the ω_{bad} region does not show improvement in x_{fix}, which is expressed as follows:

$$\mathcal{L}_3 = \max(0, s_{\omega_{\mathsf{bad}}}(x_{\mathsf{fix}}) - s_{\omega_{\mathsf{bad}}}(x_{\mathsf{bad}}))$$

For Property 4, a penalty is applied if there is excessive degradation in the $1 - \omega_{\mathsf{bad}}$ region. This is formally expressed as follows:

$$\mathcal{L}_4 = \max(0, s_{1-\omega_{\mathsf{bad}}}(x_{\mathsf{fix}}) - s_{1-\omega_{\mathsf{bad}}}(x_{\mathsf{bad}}))$$

Note that these objectives are applicable to any generative model featuring an iterative generation process, including GPT-2 [32] with chain-of-thought reasoning [44], and diffusion models utilizing denoising processes. In the next section, we will use diffusion models as a specific example.

3.3 Property-guided generation

We now outline the process for integrating the formal objective functions to guide the generation of repairs. We denote the formal guide by ϕ, as detailed in Section 3.2. During each step of the reverse diffusion process, we apply this formal guide to the sequence of continuous variables $x_{0:T}$ defined by diffusion model. Controlling $x_{0:T}$ is equivalent to sampling from the posterior $p(x_{0:T} \mid \phi) = \prod_{t=1}^{T} p(x_{t-1} \mid x_t, \phi)$, which can be decomposed into a series of guidance problems at each diffusion step as: $p(x_{t-1} \mid x_t, \phi) \propto p(x_{t-1} \mid x_t) \cdot p(\phi \mid x_{t-1}, x_t) \propto p(x_{t-1} \mid x_t) \cdot p(\phi \mid x_{t-1})$, assuming conditional independence assumptions as noted in [39]. As a result, at the t-th step, an additional gradient term for the formal guide ϕ is included on x_{t-1}: $\nabla_{x_{t-1}} \log p(x_{t-1} \mid x_t, \phi) = \nabla_{x_{t-1}} \log p(x_{t-1} \mid x_t) + \nabla_{x_{t-1}} \log p(\phi \mid x_{t-1})$, where $\log p(x_{t-1} \mid x_t)$ is provided by the underlying diffusion model trained as described in Section 3.1, and the second term is parameterized by formal guide ϕ detailed in Section 3.2. Intuitively, our framework guide the anomaly repair toward a direction that optimizes the formal properties, as visualized in Figure 3.

More specifically, we initially direct the prediction $\mu_\theta(x_t, t)$ from the approximator model towards a path that minimizes the formal objectives:

$$\hat{\mu}_\theta(x_t, t) = \mu_\theta(x_t, t) - \lambda_\phi \nabla_{x_t} \mathcal{L}(x_{\mathsf{bad}}, x_t)$$

where λ_ϕ is a hyperparameter that controls the scale of guidance. At each step x_t, the reverse process utilizes the modified $\hat{\mu}_\theta(x_t, t)$ to denoise the data, thereby obtaining x_{t-1}.

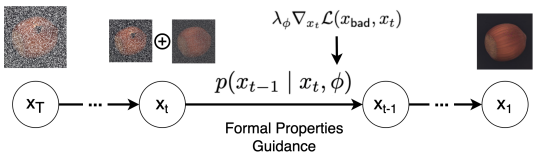


Figure 3: Property guidance on reverse diffusion steps; the gradient from the formal property loss guides the generated repair on each step.

We then add noise to the original data x_{bad} following a predefined variance schedule b_t. The noisy data at time step t, denoted as $\tilde{x}_{\text{bad},t}$ is generated by: $\tilde{x}_{\text{bad},t} = \sqrt{a_t} x_{\text{bad}} + \sqrt{1 - a_t}\mathbf{z}$, where $a_t = \Pi_{i=1}^{t}(1 - b_i)$ and $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$. Following this, we merge the normal region of $\tilde{x}_{\text{bad},t}$ with the anomalous region of x_{t-1}, in a process akin to inpainting as described in [18]:

$$\hat{x}_{t-1} = (1 - \omega) \odot \tilde{x}_{\text{bad},t} + \omega \odot x_{t-1}.$$

In summary, guidance is iteratively applied at each step of the generation process to steer the generated repair toward fulfilling the formal properties.

4 Experiments

In this section, we aim to address the following research questions:

- **RQ1 Empirical Validation**: How effectively does our framework repair the anomalies across modalities?
- **RQ2 Ablation Study**: What impact do various hyperparameters have on the performance of our framework?

4.1 Experiment Setup

Our experiments evaluate formal property-guided repairs across vision and time-series data. Each experiment employs a representative anomaly detector and dataset with predefined train-test splits. The performance of the anomaly detectors is detailed in Appendix B. A feature-wise threshold, τ_i, set at the 90% quantile of the training set's attribution values, is used to generate the anomaly map ω.

Fastflow on VisA Dataset For vision, the Fastflow Model [48] is deployed on the Visual Anomaly (VisA) Dataset [55], which comprises 12 image classes for industrial defect detection at a resolution of 512×512. We utilize the *anomalib* [4] version with a wide ResNet-50-2 backbone [50].

GPT2 on SWaT Dataset For time-series, the GPT2 [32, 54] model is applied to the Secure Water Treatment (SWaT) dataset [21]. This dataset records eleven days of operations, with seven days of normal functioning and four days of 41 simulated attacks, captured from 51 sensors and actuators. A sliding window size of 100 is used.

Baseline Diffusion Model For time-series, we employ the Diffusion-TS model [49] with 500 diffusion steps. For images, we use a diffusion model that consists of a U-Net architecture and a DDPM scheduler [10] with 1000 steps, available from the *Huggingface* library, which incorporates the concept of SDEdit [22] in its implementation. Both models are trained using the AdamW optimizer with a learning rate of 10^{-4} to generate normal outputs.

4.2 Evaluation Metrics

We evaluate the generated anomaly repairs on anomalies within the testing sets, utilizing four formal metrics as listed below. Smaller values for these metrics indicate better performance.

- **Anomaly Score of Generated Fix (M_s)**: $s(x_{\text{fix}})$
- **Difference Between Input and Generated Fix (M_d)**: $\mathbb{E}_{x_{\text{fix}}}[\|(1 - \omega) \odot (x_{\text{fix}} - x_{\text{bad}})\|_2]$
- **Degradation in Anomalous Region (M_ω)**: $\mathbb{E}_{x_{\text{fix}}}[s_\omega(x_{\text{fix}}) - s_\omega(x_{\text{bad}})]$
- **Degradation in Normal Region ($M_{1-\omega}$)**: $\mathbb{E}_{x_{\text{fix}}}[s_{(1-\omega)}(x_{\text{fix}}) - s_{(1-\omega)}(x_{\text{bad}})]$

Class	$M_s(\downarrow)$		$M_d(\downarrow)$		$M_\omega(\downarrow)$		$M_{1-\omega}(\downarrow)$	
	Baseline	Guided	Baseline	Guided	Baseline	Guided	Baseline	Guided
candle	-23.39 ± 1.23	-26.43 ± 0.14	336.06 ± 44.41	8.34 ± 0.06	-0.003 ± 0.002	-0.007 ± 0.001	0.05 ± 0.01	-0.02 ± 0.004
capsules	12.45 ± 30.02	-17.86 ± 0.20	426.69 ± 45.30	8.41 ± 0.01	0.009 ± 0.003	-0.005 ± 0.001	0.15 ± 0.02	-0.01 ± 0.002
cashew	-16.08 ± 0.85	-17.13 ± 0.33	307.39 ± 63.72	8.40 ± 0.01	-0.002 ± 0.004	-0.005 ± 0.001	0.06 ± 0.03	-0.00 ± 0.002
chewinggum	-14.56 ± 0.89	-15.70 ± 0.14	258.32 ± 53.66	8.41 ± 0.01	-0.007 ± 0.003	-0.009 ± 0.003	0.02 ± 0.02	-0.01 ± 0.002
fryum	-13.54 ± 3.02	-18.93 ± 0.42	349.18 ± 32.39	8.36 ± 0.01	0.003 ± 0.005	-0.007 ± 0.001	0.12 ± 0.04	-0.01 ± 0.003
macaroni1	199.95 ± 145.07	-25.63 ± 0.25	658.18 ± 81.65	8.41 ± 0.01	0.012 ± 0.002	-0.005 ± 0.001	0.18 ± 0.02	-0.01 ± 0.004
macaroni2	-15.61 ± 11.75	-25.18 ± 0.17	550.91 ± 55.19	8.41 ± 0.01	0.004 ± 0.003	-0.004 ± 0.001	0.11 ± 0.02	-0.01 ± 0.002
pcb1	-23.49 ± 0.49	-25.04 ± 1.08	325.54 ± 50.82	8.38 ± 0.01	-0.005 ± 0.006	-0.006 ± 0.003	0.05 ± 0.02	-0.01 ± 0.004
pcb2	-18.20 ± 0.67	-19.15 ± 0.12	289.54 ± 48.32	8.41 ± 0.01	-0.005 ± 0.003	-0.005 ± 0.001	0.03 ± 0.02	-0.01 ± 0.002
pcb3	-19.87 ± 4.29	-24.19 ± 0.15	291.16 ± 50.72	8.41 ± 0.01	-0.003 ± 0.004	-0.005 ± 0.002	0.05 ± 0.04	-0.01 ± 0.002
pcb4	2.51 ± 23.68	-20.24 ± 0.09	337.12 ± 75.48	8.41 ± 0.01	0.000 ± 0.004	-0.005 ± 0.001	0.12 ± 0.05	-0.01 ± 0.003
pipefryum	-15.08 ± 3.83	-19.80 ± 0.22	252.68 ± 78.99	8.41 ± 0.01	-0.002 ± 0.006	-0.008 ± 0.001	0.09 ± 0.05	-0.02 ± 0.005
$\Delta(\uparrow)$	**+26.54%**		**+97.46%**		**+146.42%**		**+114.16%**	

Table 1: Comparison of baseline and guided performance across four metrics for Visa dataset categories. Δ is the median improvement percentage of the guided result from baseline.

4.3 RQ1 Empirical Validation

In this section, we evaluate the performance of our framework across various domains, utilizing the anomaly masks ω generated by anomaly detectors as detailed in Section 4.1. Specifically, we compare the performance of the baseline diffusion model with our framework using four formal metrics: M_s, M_d, M_ω, and $M_{1-\omega}$. Additionally, we conduct a qualitative evaluation of the generated examples to highlight the semantic improvements.

4.3.1 Quantitative results

We compare the mean and standard deviation across testing sets for the four metrics as described in Section 4.2, and compute the improvement (Δ) of the guided results from the baseline based on the mean. The results for VisA data are presented in Table 1. For all the categories, the guided results show significant improvement over the baseline on the formal criteria, with an average improvement of 96.14%. We notice anomalous M_s values for baseline models in certain classes, which may stem from deviations in the color scheme of the generated images from their training distribution. Hence, we report the median improvement across the classes in the last row to mitigate the impact of outliers. For time-series data shown in Table 2, our guided framework achieves lower error on M_d, M_ω, and $M_{1-\omega}$, with an average improvement of 61.25% overall. The performance on M_d is not as competitive and exhibits considerable variability, likely due to the broader range of adjustments required in time-series repair to ensure continuity, in contrast to image data.

	$M_s(\downarrow)$		$M_d(\downarrow)$		$M_\omega(\downarrow)$		$M_{1-\omega}(\downarrow)$	
	Baseline	Guided	Baseline	Guided	Baseline	Guided	Baseline	Guided
SWaT	0.68 ± 0.06	0.57 ± 0.05	12.81 ± 15.82	16.28 ± 16.30	-0.029 ± 0.041	-0.094 ± 0.038	0.13 ± 0.04	0.09 ± 0.031
$\Delta(\uparrow)$	**+15.81%**		-27.07%		**+225.09%**		**+31.18%**	

Table 2: Comparison of baseline and guided performance for SWaT dataset. Δ is the improvement percentage of the guided result from baseline.

In addition, we evaluate whether the guided repairs generate normal samples by comparing them against a conformity threshold with 95% confidence derived from the training set [5]. Treating the normal class as the negative, we report the True Negative Rate (TNR) in Table 3. The TNR of most VisA categories reaches 100%, surpassing the baseline values. As a result, the guided result achieves a median TNR of 100%, which is higher than the baseline's 95.24%. For SWaT, the TNR of guided repair is 86.01%, compared to a baseline of 7.68%. On average across both domains, 92.89% of the guided repairs are classified as normal at a 95% confidence level, statistically confirming the effectiveness of AR-Pro.

Category	Baseline TNR (↑)	Guided TNR (↑)	Category	Baseline TNR (↑)	Guided TNR (↑)
Candle	**1.00**	**1.00**	Fryum	0.66	**1.00**
Capsules	0.00	**1.00**	Pipe Fryum	**1.00**	**1.00**
Cashew	**1.00**	**1.00**	PCB 1	**1.00**	0.96
Chewinggum	**1.00**	**1.00**	PCB 2	1.00	**1.00**
macaroni 1	0.11	**1.00**	PCB 3	0.90	**1.00**
macaroni 2	0.52	**1.00**	PCB 4	0.18	**1.00**
VisA Median	0.95	**1.00**	**SWaT**	0.08	**0.86**

Table 3: True Negative Rate (TNR) for VisA and SWaT.

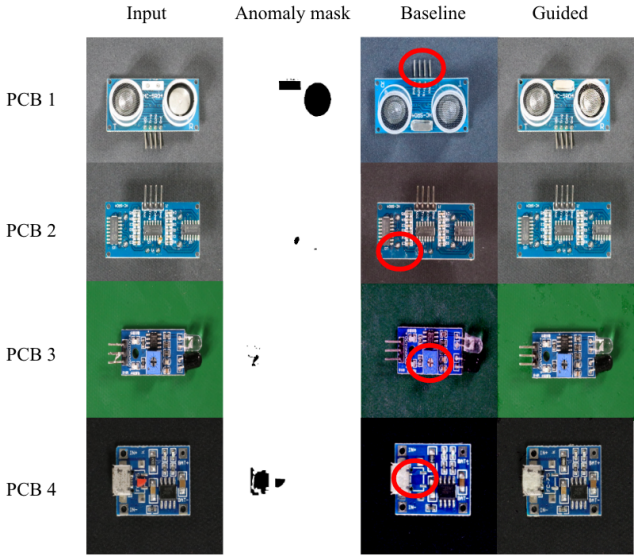


Figure 4: The original input and ground truth anomaly mask are displayed in the first two columns. The baseline method fails to preserve close similarity to the input PCB boards, as highlighted in the third column. Guided vision repair examples in the fourth column address these deficiencies.

4.3.2 Qualitative results

We present qualitative examples here to illustrate that our method can generate semantically meaningful repairs. In Figure 4, we observe that for category PCB 1, PCB 2, PCB 3 and PCB 4, the baseline fails to rigorously repair the anomaly: the orientation of the board is reversed for PCB 1; unintended white marks appear on the lower left of PCB 2; the dark lines in the potentiometer change for PCB 3; and the 'FC-75' label is missing for PCB 4. However, by integrating formal property guidance, our approach accurately reconstructs all these details while effectively removing the anomalies. This demonstrates that our method's generated repairs adhere more rigorously to the formal properties.

In Figure 5, we present examples from our time-series repair, where the anomalous time segment is shaded in red. Our results demonstrate that, guided by formal properties, AR-Pro generates signal that resemble the original signal better, as shown in the first two plots of Figure 5. In addition, we recover the sensor time-series to normal values when the baseline fails to repair the anomaly, as shown in the last two plots of Figure 5.

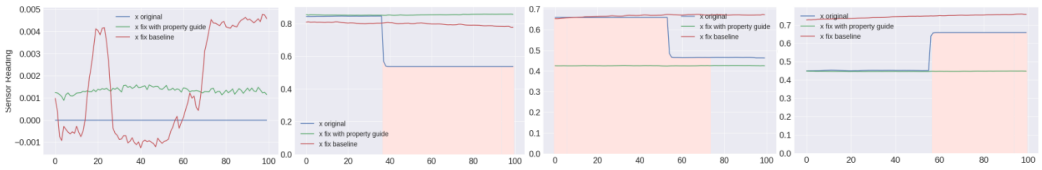
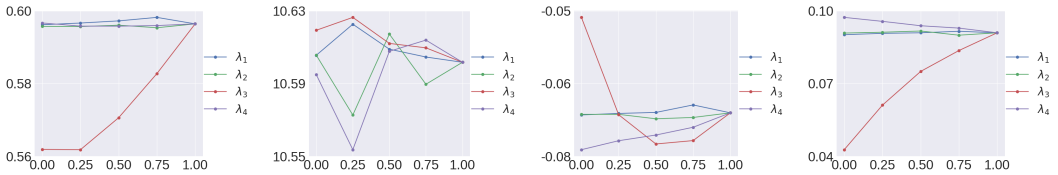


Figure 5: Original input is the blue line, property guided fix is the green line, and the baseline is the red line. The first image shows that the baseline generates a spurious signal when there is no anomaly. The second image shows that the baseline repairs the anomaly, but not as effectively as with guidance. The last two images show instances where the baseline fails to repair the anomaly.

More repair examples are available in Appendix C. However, despite the quality of the generated repairs has improved, we notice that this enhancement comes with a trade-off of increased inference time. Further details can be found in Appendix D.

4.4 RQ2 Ablation Study

We randomly sampled 100 instances to compute the median of each metric in order to evaluate the effect of hyperparameters. The parameters λ_1 to λ_4 correspond to each property. Each line in the plots represents results obtained while keeping other parameters at 1.0. The ablation results for the time series are presented in Figure 6, with additional plots for λ_ϕ available in Appendix E.



(a) M_s (b) M_d (c) M_ω (d) $M_{1-\omega}$

Figure 6: Effects of hyperparameters on M_s, M_d, M_ω and $M_{1-\omega}$; the range of change remains small.

Our observations indicate that variations in the scale of property hyperparameters do not significantly impact the formal metrics, as the range of change remains relatively small. In addition, no consistent trends were observed when varying the λ parameters across the metrics. This suggests that our framework demonstrates robust performance, and extensive parameter tuning may be unnecessary.

5 Related Works

Anomaly detection is essential across various domains, with numerous techniques developed. Traditional statistical methods, such as ARIMA [24] and Gaussian models [34], struggle with high-dimensional data. In contrast, deep learning methods, including autoencoders [13] and GANs [9], can detect high-dimensional anomalies via reconstruction error. For time-series data, LSTMs [17] and transformer-based models [41, 45, 51, 54] have shown exceptional performance. Additionally, diffusion models are emerging as promising tools for visual anomaly detection [25, 53]. While these methods vary in strengths and are continually improving, explaining anomalies presents significant challenges. Most current methods rely on feature importance scores or visualizations [26], such as gradients or reconstructions [28], which often fail to provide actionable insights. The lack of formal frameworks [47] and consistent evaluation metrics [1] complicates this issue. For example, the absence of formal metrics leads to inconsistencies in evaluation [12], underscoring the need for more rigorous approaches and reliable criteria [20].

Our work also intersects with guided generation across various domains. Leading this paradigm are Denoising Diffusion Probabilistic Models (DDPMs) [10, 22], enhancing image realism by adding noise to a guided input and then denoising it. For time-series, Diffusion-TS [49] generates multivariate time series through an encoder-decoder transformer. However, our formal framework, which is specialized for anomaly repair, is applicable to any generative model that utilizes an iterative generation process.

6 Discussion

We note that counterfactuals may seem unnecessary for datasets like VisA and SWaT, where normal patterns are clearly defined and an anomaly map may suffice. However, rigorous counterfactuals are necessary in complex scenarios such as medical imaging [40], where patient variability is significant, resolving anomalies is not straightforward. In addition, we remark that the quality of repairs also depends on the performance of both the anomaly detector and the region selector, identifying regions requiring correction. Moreover, to obtain a guaranteed repair, one can can use statistical tools like conformal prediction [36] together with our framework: continue generating repairs until the anomaly score falls below the conformity threshold, ensuring the anomaly is repaired.

7 Conclusion

Our AR-Pro framework complements the anomaly detection field by emphasizing counterfactual explanation and repair. We leverage feature-level attributions to effectively identify and rectify anomalies. Key contributions include the establishment of universal properties for anomaly repairs and the development of property-guided generation process that produces high-quality repairs, which outperform the off-the-shelf diffusion models by 61.25% on SWaT dataset and 96.14% on VisA dataset. We also demonstrate the semantic meaningfulness of these repairs with examples. In sum, our framework, AR-Pro, presents a versatile, effective solution for both explaining anomalies and synthesizing repairs across vision and time-series modalities.

References

[1] C. Agarwal, S. Krishna, E. Saxena, M. Pawelczyk, N. Johnson, I. Puri, M. Zitnik, and H. Lakkaraju. Openxai: Towards a transparent evaluation of model explanations. *Advances in Neural Information Processing Systems*, 35:15784–15799, 2022.

[2] C. M. Ahmed, V. R. Palleti, and A. P. Mathur. Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks*, pages 25–28, 2017.

[3] M. Ahmed, A. N. Mahmood, and M. R. Islam. A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, 55:278–288, 2016.

[4] S. Akcay, D. Ameln, A. Vaidya, B. Lakshmanan, N. Ahuja, and U. Genc. Anomalib: A deep learning library for anomaly detection. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 1706–1710. IEEE, 2022.

[5] A. N. Angelopoulos and S. Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

[6] K. Batzner, L. Heckler, and R. König. Efficientad: Accurate visual anomaly detection at millisecond-level latencies. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 128–138, 2024.

[7] P. Bergmann, M. Fauser, D. Sattlegger, and C. 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.

[8] Z. Chen, G. Xu, V. Mahalingam, L. Ge, J. Nguyen, W. Yu, and C. Lu. A cloud computing based network monitoring and threat detection system for critical infrastructures. *Big Data Research*, 3:10–23, 2016.

[9] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[10] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[11] A. Høst-Madsen, E. Sabeti, and C. Walton. Data discovery and anomaly detection using atypicality: Theory. *IEEE Transactions on Information Theory*, 65(9):5302–5322, 2019.

[12] J. Jiang, F. Leofante, A. Rago, and F. Toni. Formalising the robustness of counterfactual explanations for neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[13] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[14] C.-Y. Lai, F.-K. Sun, Z. Gao, J. H. Lang, and D. S. Boning. Nominality score conditioned time series anomaly detection by point/sequential reconstruction. *arXiv preprint arXiv:2310.15416*, 2023.

[15] X. A. Larriva-Novo, M. Vega-Barbas, V. A. Villagrá, and M. S. Rodrigo. Evaluation of cybersecurity data set characteristics for their applicability to neural networks algorithms detecting cybersecurity anomalies. *IEEE Access*, 8:9005–9014, 2020.

[16] X. Li, J. Thickstun, I. Gulrajani, P. S. Liang, and T. B. Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

[17] B. Lindemann, B. Maschler, N. Sahlab, and M. Weyrich. A survey on anomaly detection for technical systems using lstm networks. *Computers in Industry*, 131:103498, 2021.

[18] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11461–11471, 2022.

[19] A. Manolache, F. Brad, and E. Burceanu. Date: Detecting anomalies in text via self-supervision of transformers. *arXiv preprint arXiv:2104.05591*, 2021.

[20] J. Marques-Silva and A. Ignatiev. Delivering trustworthy ai through formal xai. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[21] A. P. Mathur and N. O. Tippenhauer. Swat: A water treatment testbed for research and training on ics security. In *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*, pages 31–36. IEEE, 2016.

[22] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

[23] A. Moallemi, A. Burrello, D. Brunelli, and L. Benini. Exploring scalable, distributed real-time anomaly detection for bridge health monitoring. *IEEE Internet of Things Journal*, 9(18):17660–17674, 2022.

[24] H. Z. Moayedi and M. Masnadi-Shirazi. Arima model for network traffic prediction and anomaly detection. In *2008 international symposium on information technology*, volume 4, pages 1–6. IEEE, 2008.

[25] A. Mousakhan, T. Brox, and J. Tayyub. Anomaly detection with conditioned denoising diffusion models. *arXiv preprint arXiv:2305.15956*, 2023.

[26] M. Nauta, J. Trienes, S. Pathak, E. Nguyen, M. Peters, Y. Schmitt, J. Schlötterer, M. van Keulen, and C. Seifert. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. *ACM Computing Surveys*, 55(13s):1–42, 2023.

[27] G. Pallotta, M. Vespe, and K. Bryan. Vessel pattern knowledge discovery from ais data: A framework for anomaly detection and route prediction. *Entropy*, 15(6):2218–2245, 2013.

[28] G. Pang and C. Aggarwal. Toward explainable deep anomaly detection. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 4056–4057, 2021.

[29] G. Pang, C. Ding, C. Shen, and A. v. d. Hengel. Explainable deep few-shot anomaly detection with deviation networks. *arXiv preprint arXiv:2108.00462*, 2021.

[30] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel. Deep learning for anomaly detection: A review. *ACM computing surveys (CSUR)*, 54(2):1–38, 2021.

[31] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo. Fraud detection: A systematic literature review of graph-based anomaly detection approaches. *Decision Support Systems*, 133:113303, 2020.

[32] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[33] D. Ramotsoela, A. Abu-Mahfouz, and G. Hancke. A survey of anomaly detection in industrial wireless sensor networks with critical water system infrastructure as a case study. *Sensors*, 18(8):2491, 2018.

[34] P. J. Rousseeuw and M. Hubert. Anomaly detection by robust statistics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(2):e1236, 2018.

[35] K. K. Santhosh, D. P. Dogra, and P. P. Roy. Anomaly detection in road traffic using visual surveillance: A survey. *ACM Computing Surveys (CSUR)*, 53(6):1–26, 2020.

[36] G. Shafer and V. Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

[37] H.-K. Shin, W. Lee, J.-H. Yun, and H. Kim. {HAI} 1.0:{HIL-based} augmented {ICS} security dataset. In *13Th USENIX workshop on cyber security experimentation and test (CSET 20)*, 2020.

[38] K. Sokol and P. Flach. Counterfactual explanations of machine learning predictions: Opportunities and challenges for ai safety. In *2019 AAAI Workshop on Artificial Intelligence Safety, SafeAI 2019*. CEUR Workshop Proceedings, 2019.

[39] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[40] J. J. Thiagarajan, K. Thopalli, D. Rajan, and P. Turaga. Training calibration-based counterfactual explainers for deep learning models in medical image analysis. *Scientific reports*, 12(1):597, 2022.

[41] S. Tuli, G. Casale, and N. R. Jennings. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv preprint arXiv:2201.07284*, 2022.

[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[43] S. Verma, J. Dickerson, and K. Hines. Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*, 2, 2020.

[44] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[45] J. Xu, H. Wu, J. Wang, and M. Long. Anomaly transformer: Time series anomaly detection with association discrepancy. *arXiv preprint arXiv:2110.02642*, 2021.

[46] V. Yepmo, G. Smits, and O. Pivert. Anomaly explanation: A review. *Data & Knowledge Engineering*, 137:101946, 2022.

[47] J. Yu, A. Ignatiev, P. J. Stuckey, N. Narodytska, and J. Marques-Silva. Eliminating the impossible, whatever remains must be true: On extracting and applying background knowledge in the context of formal explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[48] J. Yu, Y. Zheng, X. Wang, W. Li, Y. Wu, R. Zhao, and L. Wu. Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. *arXiv preprint arXiv:2111.07677*, 2021.

[49] X. Yuan and Y. Qiao. Diffusion-ts: Interpretable diffusion for general time series generation. *arXiv preprint arXiv:2403.01742*, 2024.

[50] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[51] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2114–2124, 2021.

[52] A. Zhang, S. Song, J. Wang, and P. S. Yu. Time series data cleaning: From anomaly detection to anomaly repairing. *Proceedings of the VLDB Endowment*, 10(10):1046–1057, 2017.

[53] H. Zhang, Z. Wang, Z. Wu, and Y.-G. Jiang. Diffusionad: Denoising diffusion for anomaly detection. *arXiv preprint arXiv:2303.08730*, 2023.

[54] T. Zhou, P. Niu, L. Sun, R. Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36, 2024.

[55] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In *European Conference on Computer Vision*, pages 392–408. Springer, 2022.

A Computational Configuration

All experiments were conducted on three NVIDIA GeForce RTX 4090, each equipped with 24 GB of memory and 16,384 CUDA cores.

B Anomaly Detector Performance

In this section, we report the performance of anomaly detectors. The results for FastFlow [48] are presented in Table 4, and the results for GPT-2 [54] are shown in Table 5. We did not perform extensive parameter tuning, as the performance of anomaly detectors are not the primary focus of our work. In addition, it is recommended for users adhere to usage guidelines of using GPT2 or implementing safety filters.

Category	Image AUROC	Pixel AUROC
Candle	0.6501	0.5617
Capsules	0.5352	0.8188
Cashew	0.3816	0.7558
Chewing Gum	0.3754	0.9362
Fryum	0.761	0.4244
Macaroni1	0.2607	0.9127
Macaroni2	0.504	0.7729
PCB1	0.7173	0.9163
PCB2	0.7398	0.7945
PCB3	0.5361	0.8073
PCB4	0.6331	0.6183
Pipe Fryum	0.4696	0.2799
Average	0.5470	0.7166

Table 4: AUROC Scores of Fastflow for Various Categories in VisA

Metric	GPT2 on SWaT
Accuracy	0.9784
Precision	0.8831
Recall	0.9472
F1-score	0.9140

Table 5: Performance of GPT2 anomaly detectors on SWaT Datasets

C More Qualitative Examples

More VisA categories examples can be found in Figure 7.

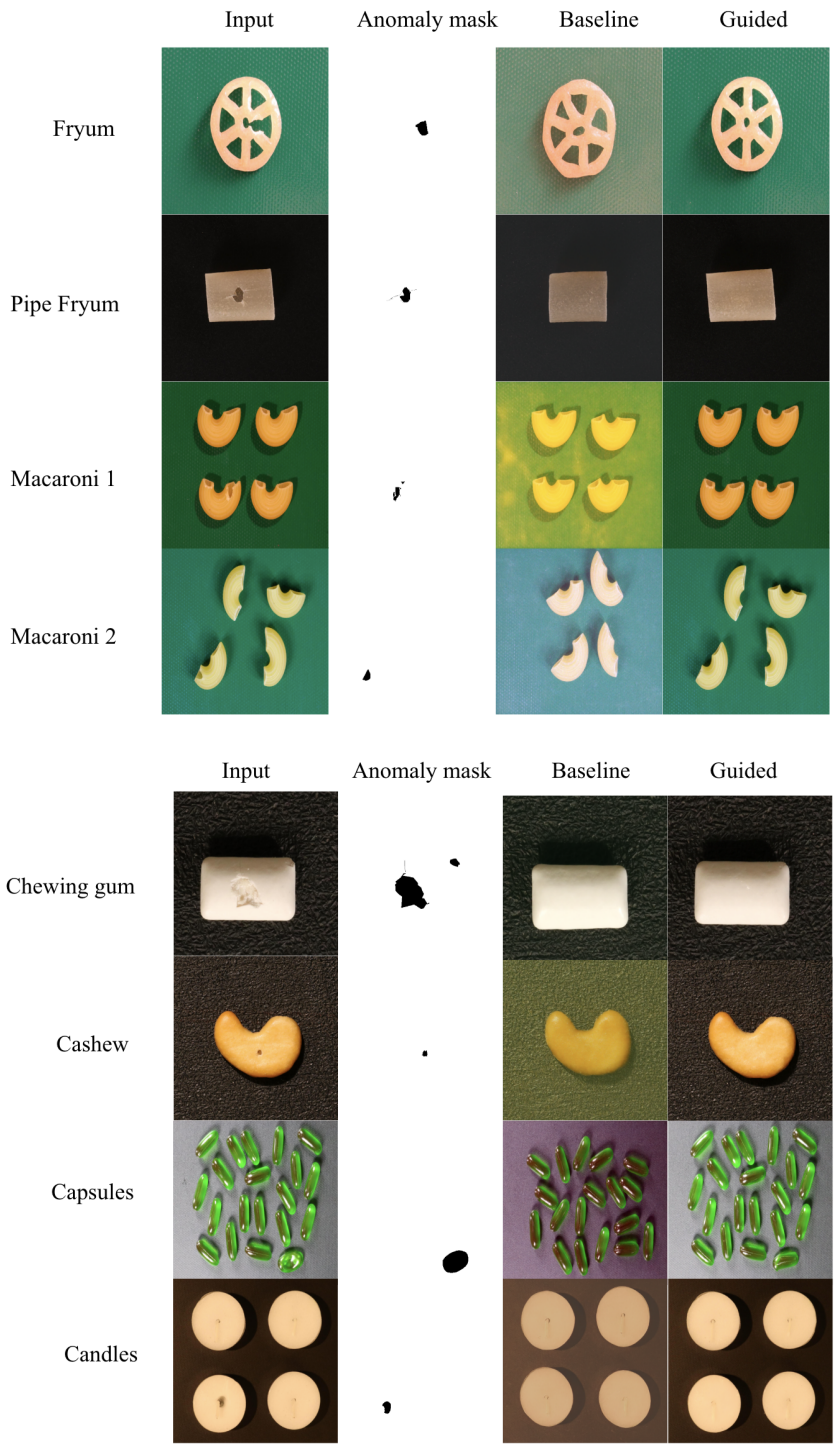
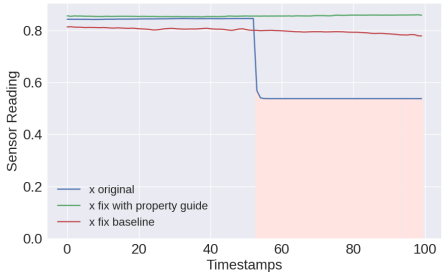
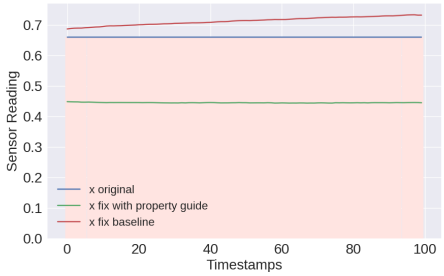


Figure 7: More VisA examples. With AR-Pro, we have anomaly repairs resemble inputs better, compared with baseline.

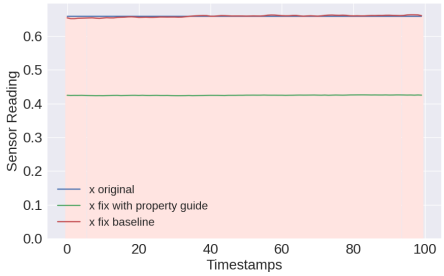
More time-series examples can be found in Figure 8.



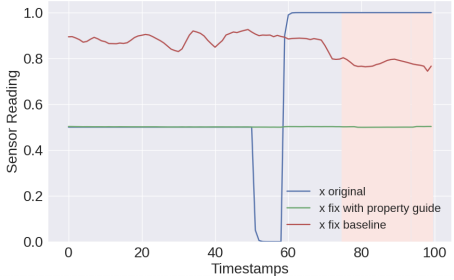
(a) Both repair anomaly.



(b) Baseline fails to repair anomaly.



(c) Baseline fails to repair anomaly.



(d) Baseline fails to repair anomaly.

Figure 8: The first image shows an instance where the baseline repairs the anomaly, but not as effectively as with property guidance. The last three images show instances where the baseline fails to repair the anomaly.

D Inference time

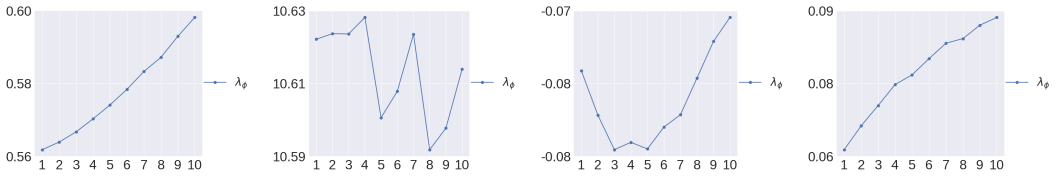
We randomly sampled 50 instances from the testing set to compare inference times, as shown in Table 6. The median values, measured in seconds, compare the baseline with our framework. We observed that although property guidance generates higher-quality repairs, it results in slightly longer inference times for time series data and significantly longer times for image data, possibly due to the higher dimensionality of the inputs. In total, it took about 40 hours to finish RQ 1 for VisA and 25 hours to finish SWaT. For RQ 2, it took 5 hours for SWaT. Improving computation time will be a focus of our future work.

Vision		Time-series	
Baseline	Guided	Baseline	Guided
126.50	236.37	10.86	12.49

Table 6: Comparison of Baseline and Guided Generation Time (in s)

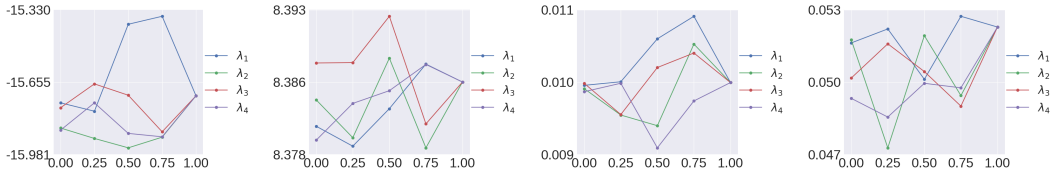
E More Ablation Plots

The ablation study for λ_ϕ on time series can be found in Figure 9. The ablation for image data, using cashew class as an example can be found in Figure 10 and Figure 11.



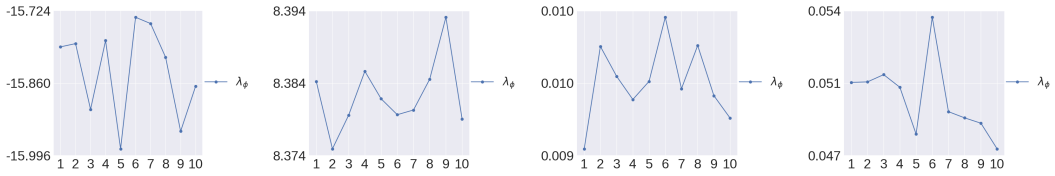
(a) M_s (b) M_d (c) M_ω (d) $M_{1-\omega}$

Figure 9: Effects of hyperparameter λ_ϕ on M_s, M_d, M_ω and $M_{1-\omega}$ on SWaT; the effect of λ_ϕ varies across metrics, but the range remain relatively small.



(a) M_s (b) M_d (c) M_ω (d) $M_{1-\omega}$

Figure 10: Effects of hyperparameter on M_s, M_d, M_ω and $M_{1-\omega}$ on VisA cashew class; the effect of λ_ϕ varies across metrics, but the range remain relatively small.



(a) M_s (b) M_d (c) M_ω (d) $M_{1-\omega}$

Figure 11: Effects of hyperparameter λ_ϕ on M_s, M_d, M_ω and $M_{1-\omega}$ on VisA cashew class; the effect of λ_ϕ varies across metrics, but the range remain relatively small.

We include the more recent vision anomaly detector, EfficientAD [6], and an additional dataset, MVTec [7]. The results for the Fastflow model are presented in Table 7 , and those for the EfficientAD model are shown in Table 8. We observe that even with the changes in the anomaly detector and dataset, there is still a considerable improvement across the four metrics with our formal property guidance.

Class	$M_s(\downarrow)$		$M_d(\downarrow)$		$M_\omega(\downarrow)$		$M_{1-\omega}(\downarrow)$	
	Baseline	Guided	Baseline	Guided	Baseline	Guided	Baseline	Guided
transistor	-12.15 ± 0.16	-13.81 ± 0.17	265.05 ± 75.78	7.04 ± 1.96	-0.006 ± 0.002	-0.006 ± 0.001	0.02 ± 0.01	-0.01 ± 0.002
$\Delta(\uparrow)$	**+26.54%**		**+97.46%**		**+146.42%**		**+114.16%**	

Table 7: Comparison of baseline and guided performance across four metrics for MVTec dataset categories with FastFlow. Δ is the median improvement percentage of the guided result from baseline.

Class	$M_s(\downarrow)$		$M_d(\downarrow)$		$M_\omega(\downarrow)$		$M_{1-\omega}(\downarrow)$	
	Baseline	Guided	Baseline	Guided	Baseline	Guided	Baseline	Guided
cable	6.61 ± 0.96	6.27 ± 0.63	379.84 ± 54.00	9.15 ± 1.08	-0.101 ± 0.052	-0.138 ± 0.026	0.34 ± 0.06	-0.03 ± 0.009
capsule	141.28 ± 3.57	142.71 ± 2.68	223.80 ± 27.91	10.59 ± 1.18	-0.383 ± 0.396	-0.140 ± 0.223	-0.28 ± 0.34	-0.07 ± 0.088
carpet	6.57 ± 3.61	0.70 ± 0.03	413.55 ± 54.08	10.16 ± 1.38	0.344 ± 0.258	-0.011 ± 0.004	2.23 ± 1.58	-0.00 ± 0.001
hazelnut	95.76 ± 9.10	49.35 ± 17.89	244.18 ± 26.76	9.72 ± 0.95	2.348 ± 1.048	-0.875 ± 0.787	21.10 ± 2.31	0.04 ± 0.065
leather	36.89 ± 1.87	16.41 ± 0.63	481.89 ± 68.35	10.00 ± 1.37	1.339 ± 0.274	-0.234 ± 0.190	11.84 ± 1.15	0.01 ± 0.016
metal nut	3.52 ± 0.09	2.58 ± 0.13	422.73 ± 50.43	10.40 ± 1.27	0.021 ± 0.035	-0.027 ± 0.036	0.63 ± 0.06	-0.00 ± 0.002
pill	4.54 ± 0.09	2.38 ± 0.11	261.34 ± 14.31	10.88 ± 0.73	0.175 ± 0.025	-0.015 ± 0.021	0.54 ± 0.04	-0.01 ± 0.005
transistor	37.15 ± 3.98	45.77 ± 11.95	272.63 ± 61.83	44.72 ± 140.20	-1.385 ± 0.708	-0.403 ± 0.662	-1.86 ± 2.31	-0.62 ± 2.089
$\Delta(\uparrow)$	**+31.07%**		**+95.17%**		**+99.10%**		**+94.01%**	

Table 8: Comparison of baseline and guided performance across four metrics for MVTec dataset categories with EfficientAD as the anomaly detector. Δ is the median improvement percentage of the guided result from baseline.

For the time-series experiments, we add a new anomaly detector, Llama 2, and two time-series anomaly detection datasets, WADI [2] and HAI [37], as shown in Table 9. We also report the performance of GPT-2 in Table 10. Despite the changes in the anomaly detector and datasets, our formal property guidance continues to achieve significant improvements across the four metrics.

Dataset	$M_s(\downarrow)$		$M_d(\downarrow)$		$M_\omega(\downarrow)$		$M_{1-\omega}(\downarrow)$	
	Baseline	Guided	Baseline	Guided	Baseline	Guided	Baseline	Guided
SWaT	0.83 ± 0.05	0.59 ± 0.04	3.05 ± 0.61	7.25 ± 2.50	0.084 ± 0.026	-0.026 ± 0.008	0.19 ± 0.02	0.05 ± 0.012
WADI	0.83 ± 0.05	0.59 ± 0.04	3.05 ± 0.61	7.25 ± 2.50	0.084 ± 0.026	-0.026 ± 0.008	0.19 ± 0.02	0.05 ± 0.012
HAI	0.83 ± 0.05	0.59 ± 0.04	3.05 ± 0.61	7.25 ± 2.50	0.084 ± 0.026	-0.026 ± 0.008	0.19 ± 0.02	0.05 ± 0.012
$\Delta(\uparrow)$	**+26.54%**		**+97.46%**		**+146.42%**		**+114.16%**	

Table 9: Comparison of baseline and guided performance across four metrics for SWaT, WADI, and HAI dataset categories with llama2 model. Δ is the median improvement percentage of the guided result from baseline.

Dataset	$M_s(\downarrow)$		$M_d(\downarrow)$		$M_\omega(\downarrow)$		$M_{1-\omega}(\downarrow)$	
	Baseline	Guided	Baseline	Guided	Baseline	Guided	Baseline	Guided
SWaT	0.68 ± 0.06	0.57 ± 0.05	12.81 ± 15.82	16.28 ± 16.30	-0.029 ± 0.041	-0.094 ± 0.038	0.13 ± 0.04	0.09 ± 0.031
WADI	0.83 ± 0.05	0.59 ± 0.04	3.05 ± 0.61	7.25 ± 2.50	0.084 ± 0.026	-0.026 ± 0.008	0.19 ± 0.02	0.05 ± 0.012
HAI	0.83 ± 0.05	0.59 ± 0.04	3.05 ± 0.61	7.25 ± 2.50	0.084 ± 0.026	-0.026 ± 0.008	0.19 ± 0.02	0.05 ± 0.012
$\Delta(\uparrow)$	**+26.54%**		**+97.46%**		**+146.42%**		**+114.16%**	

Table 10: Comparison of baseline and guided performance across four metrics for SWaT, WADI, and HAI dataset categories with GPT-2 model. Δ is the median improvement percentage of the guided result from baseline.