AUGMENTING VISUAL ANOMALY DETECTION WITH AUTOMATED INTERPRETABILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual anomaly detectors identify deviations from known-normal data, but their anomaly signals may mix evidence of actual anomalies with benign visual variation. We investigate whether automated interpretability can augment visual anomaly detectors by identifying and intervening on different components of this signal. We decompose PatchCore nearest-normal residuals into sparse features using Sparse Autoencoders (SAEs), and provide high-activation and contrastive non-active examples to a Multimodal LLM, which describes each feature and labels it as anomaly, distractor, or uncertain. These labels guide interventions in the SAE hidden representation, where distractor features are suppressed and anomaly features amplified. The edited representation is then used to reconstruct patch embeddings, which are rescored with PatchCore. Across 40 categories from four benchmarks, applying both interventions jointly improves macro-average image-level AUROC from 0.8724 to 0.8857 on source data and from 0.8066 to 0.8210 under synthetic corruptions. On three additional RobustAD categories with real acquisition shifts, the same interventions improve AUROC from 0.8745 to 0.9056 on source data and from 0.6069 to 0.6599 under real acquisition shifts. Finally, individual feature interventions across all 43 categories show that the MLLM labels are aligned in aggregate with how features differently affect normal and anomalous images. Our code is available at `https://anonymous.4open.science/r/vad-interp-EFE8`.

1 INTRODUCTION

Visual anomaly detection aims to identify anomalies without requiring examples of all possible anomaly types. Methods such as PatchCore (Roth et al., 2022) approach this problem by comparing representations of an input image with representations extracted from known-normal data. However, not every difference from known-normal examples in embedding space reflects a defect or a violation of the expected object configuration. Benign visual changes, such as differences in background, lighting, viewpoint, surface appearance, or manufacturing variations that still fall within the definition of normality, can also move an input representation away from its normal reference. The resulting signal may therefore mix evidence of actual anomalies with variation that should ideally not affect the detector's decision. Recent work in mechanistic interpretability has used Sparse Autoencoders (SAEs) to decompose neural representations into sparse features (Bricken et al., 2023; Gao et al., 2025), and language or multimodal models to generate human-understandable descriptions of these features (Templeton et al., 2024; De Santis et al., 2026b). This motivates the question of whether these automated interpretations can separate task-relevant anomaly evidence from distracting variation well enough to guide interventions on the detector itself.

In this paper, we investigate this question by decomposing the anomaly residuals of a frozen visual anomaly detector into sparse features and automatically interpreting what these features represent. Using a small calibration set containing both normal and anomalous images, we compute residuals between patch embeddings and their nearest known-normal embeddings and decompose these residuals with an SAE. We then provide high-activation and contrastive examples of each learned feature to a Multimodal Large Language Model (MLLM), which describes the visual pattern associated with the feature and labels it as *anomaly*, *distractor*, or *uncertain*. These labels are used to suppress distractor features and amplify anomaly features. The resulting patch embeddings are then rescored by the same frozen anomaly detector. Features labeled as *uncertain* are left unchanged.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We organize our study around three main research questions. **RQ1** asks whether feature interventions guided by automated interpretability improve anomaly detection performance on source data. **RQ2** asks whether these interventions remain effective under synthetic corruptions and real acquisition shifts. **RQ3** asks to what extent the labels assigned by the MLLM correspond to the effects of suppressing or amplifying individual SAE features in isolation. Our results show that the proposed feature interventions improve PatchCore performance on source data across 40 categories from four visual anomaly detection benchmarks. These gains persist under synthetic corruptions. We also evaluate three additional categories from RobustAD, a benchmark with real acquisition shifts, where the interventions produce the largest improvements. Finally, individual feature interventions across all 43 categories show that the MLLM labels are aligned in aggregate with how the corresponding features differently affect normal and anomalous images.

2 Related Work

Mechanistic Interpretability. Mechanistic interpretability aims to reverse-engineer neural networks by understanding the roles of their internal components, such as neurons and their interactions. Directly interpreting neurons is challenging because they often do not map one-to-one to human-understandable concepts. A single neuron can respond to multiple unrelated concepts (Bau et al., 2017), a phenomenon known as polysemanticity (Olah et al., 2020). A prominent explanation for polysemanticity is superposition, where a network represents more features than there are available activation dimensions by encoding them along non-orthogonal directions (Elhage et al., 2022). One way to avoid relying on individual neurons is to represent concepts as directions in activation space, which can be constructed from user-provided examples (Kim et al., 2018; De Santis et al., 2026a). Related work has shown that directions in activation space can also be used to steer model behavior (Rimsky et al., 2024). Concept discovery methods can instead identify these concept directions directly by applying clustering or factorization techniques in the model activation space (Ghorbani et al., 2019; Zhang et al., 2021; Fel et al., 2023; Bianchi et al., 2026). Sparse Autoencoders (SAEs) extend this idea by learning a sparse dictionary of features from model activations with an autoencoder, with only a small number of features active for each input, providing a way to recover features represented in superposition (Bricken et al., 2023).

Automated interpretability refers to using language or multimodal models to automatically map these discovered features to human-understandable concepts. Early work used language models to describe neurons from their own activation examples (Bills et al., 2023), while subsequent work extended this approach to vision models (Rott Shaham et al., 2024). De Santis et al. (2026b) further combined SAEs with MLLMs to extract and automatically label concepts learned by vision backbones. Sparse features can also provide an interface for editing internal representations (Bricken et al., 2023; Marks et al., 2025). In our setting, we use an MLLM to label sparse SAE features that decompose the anomaly signal of a visual anomaly detector, and use these labels to automatically determine which features should be suppressed or amplified.

Visual Anomaly Detection. Visual anomaly detection typically models normal data and identifies inputs that deviate from it, avoiding the assumption that all relevant anomaly types are observed during training. Existing approaches model normality in different ways, including feature distributions (Defard et al., 2021; Gudovskiy et al., 2022), stored normal representations (Roth et al., 2022), and discrepancies between learned representations (Deng & Li, 2022; Batzner et al., 2024; Ndiour et al., 2023). PatchCore (Roth et al., 2022), which we use in our main experiments, stores normal patch embeddings in a memory bank and scores test images using their distance to these embeddings. FRE (Ndiour et al., 2023), which we use as an alternative detector in Appendix E, instead models normality using a shallow linear autoencoder and scores anomalies from its reconstruction error. In terms of interpretability, prior work has primarily focused on local explanations and spatial localization of anomalous regions (Wang et al., 2025; Liznerski et al., 2021; Jiang et al., 2023). In contrast, we decompose the anomaly signal into a global dictionary of features, assign them semantic interpretations, and use these interpretations to guide interventions on the detector.

One prominent approach to evaluating robustness in these visual anomaly detectors is to apply synthetic image corruptions at multiple severity levels (Hendrycks & Dietterich, 2019; Wang et al., 2024). Such benchmarks apply controlled perturbations to the source images, such as noise, blur, and contrast variations. Pemula et al. (2025) later introduced RobustAD, a benchmark to instead

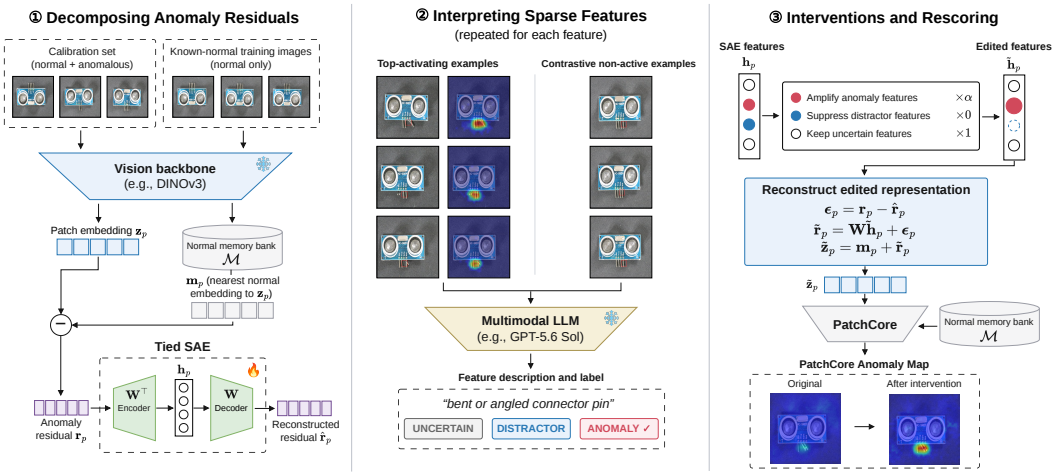


Figure 1: Overview of the pipeline. (1) Given a frozen vision backbone and a PatchCore memory bank, we compute anomaly residuals relative to the nearest normal patch embeddings and decompose them using a tied Sparse Autoencoder (SAE). (2) A multimodal LLM is prompted with top-activating and contrastive non-active examples to generate SAE feature descriptions and assign an anomaly, distractor, or uncertain label. (3) We amplify anomaly features and suppress distractor ones. Then we reconstruct the edited patch embeddings and rescore them with PatchCore.

investigate robustness in real-world scenarios. To do so, they collected real images under controlled variations in lighting, object pose, and background, and showed that synthetic corruptions do not fully reflect robustness to real acquisition changes. For this reason, we evaluate our interventions under both synthetic corruptions and the real acquisition shifts of RobustAD.

3 Methodology

Our methodology consists of three steps illustrated in Figure 1. We decompose deviations from normality into sparse features using a Sparse Autoencoder (SAE), interpret these features using a Multimodal Large Language Model (MLLM), and intervene on the interpreted features before rescoring the edited representations with the same frozen anomaly detector.

Decomposing Anomaly Residuals. Our starting point is a frozen visual anomaly detector that compares the representations of an input image against a known representation of normality. In our main experiments, we instantiate this setting using PatchCore (Roth et al., 2022). Given an input image $\boldsymbol{x}$, a frozen backbone ϕ produces a set of patch embeddings $\mathbf{z}_p \in \mathbb{R}^d$, while PatchCore stores embeddings extracted from known-normal training images in a dataset-specific memory bank $\mathcal{M}$. For each patch p, PatchCore retrieves its nearest normal embedding:

$$\mathbf{m}_p = \arg\min_{\mathbf{m}\in\mathcal{M}} \|\mathbf{z}_p - \mathbf{m}\|_2 \tag{1}$$

The nearest-neighbor distance $\|\mathbf{z}_p - \mathbf{m}_p\|_2$ provides the magnitude of the patch-level anomaly evidence that is subsequently aggregated by the PatchCore scoring rule. We define the associated deviation in the embedding space as the following residual term:

$$\mathbf{r}_p = \mathbf{z}_p - \mathbf{m}_p \in \mathbb{R}^d \tag{2}$$

We train a Sparse Autoencoder (SAE) on these patch residuals computed from a balanced calibration set containing normal and anomalous images, with all calibration residuals defined relative to the same normal memory bank $\mathcal{M}$, as shown in step ① of Figure 1. Although our main experiments use PatchCore, the method only requires an anomaly detector that defines a deviation in embedding space from a normal reference. We ablate the choice of anomaly detector in Appendix E by replacing PatchCore with FRE (Ndiour et al., 2023).

The SAE uses TopK sparsity, which directly controls the number of active features and has been shown to provide a Pareto improvement over L_1 regularization in the sparsity and reconstruction

error trade-off (Gao et al., 2025). We additionally tie encoder and decoder weights and do not use biases. Following Bricken et al. (2023), we also constrain the dictionary vectors to unit L_2 norm. Together with the ReLU, these constraints ensure that, all else equal, an active feature makes the residual larger in magnitude than it would be if that feature were inactive. We do this to encourage a dictionary of features that account for deviations from normality, rather than encoding normality itself. In Appendix B, we compare this configuration with an untied TopK SAE with learned biases and observe comparable reconstruction error and recovered anomaly detection performance.

Let $\mathbf{W} \in \mathbb{R}^{d \times m}$ denote the shared encoder-decoder weight matrix. For patch p, the sparse hidden representation and reconstructed residual are:

$$\mathbf{h}_p = \mathrm{TopK}_k \left(\mathrm{ReLU} \left(\mathbf{W}^\top \mathbf{r}_p \right) \right), \qquad \hat{\mathbf{r}}_p = \mathbf{W}\mathbf{h}_p \tag{3}$$

where TopK_k retains the k largest activations and sets all remaining entries to zero.

We minimize mean squared reconstruction error over the calibration residuals. Let $\mathcal{R}$ denote the collection of calibration residuals. Since sparsity is imposed directly by the TopK operator, no additional sparsity penalty is used:

$$\mathcal{L}_{\mathrm{SAE}} = \frac{1}{|\mathcal{R}|d} \sum_{\mathbf{r} \in \mathcal{R}} \|\mathbf{r} - \hat{\mathbf{r}}\|_2^2 \tag{4}$$

Interpreting Sparse Features. We next interpret the sparse features learned by the SAE using a Multimodal Large Language Model (MLLM), as illustrated in step ② of Figure 1. For each feature, we select calibration patches with the highest activations, taking at most one patch from each image. For MVTec AD 2, this constraint is applied at the scene level. For each selected patch, the MLLM receives the full image with the corresponding PatchCore region highlighted by a red box, a spatial SAE feature activation map, and a close-up of the selected region and its surroundings. Activation maps for the same feature use a common scale, normalized by that feature's maximum activation over the calibration set. We additionally provide contrastive examples of patches where the feature has zero activation. To emphasize what distinguishes feature activation from non-activation, we select zero-activation patches whose patch embeddings $\mathbf{z}_p$ are most similar to the mean embedding of the selected high-activation patches, using cosine similarity. For zero-activation examples, we likewise select at most one patch from each image, or from each scene for MVTec AD 2. Each contrastive example shows the full image with the selected region highlighted and a close-up of the region. The MLLM also receives known-normal images for the corresponding category, arranged in a 3×3 grid, together with the dataset and category context. For MVTec LOCO AD and RobustAD, we additionally provide the benchmark description of the expected normal configuration.

Given this evidence, the MLLM produces a concise description of the visual pattern associated with the feature and assigns one of three labels. It is instructed to use *anomaly* when the feature indicates an anomaly of the inspected object or its expected configuration, *distractor* when it captures nuisance variation, visual artifacts, or ordinary variation in the appearance of the object or scene, and *uncertain* when the evidence is mixed or unclear. If either the high-activation or zero-activation group contains fewer than two distinct examples after the one-per-image (or one-per-scene) selection, the feature is assigned *uncertain* without querying the MLLM. An example of a complete prompt is provided in Appendix H.

Interventions and Rescoring. Finally, we use the labels assigned to the SAE features to define the interventions, as illustrated in step ③ of Figure 1. We consider three interventions: distractor suppression, anomaly amplification, and their combination, which we refer to as joint intervention. For distractor suppression, we zero-ablate features labeled as *distractor* by setting their activations to zero, while leaving other features unchanged. For anomaly amplification, we multiply the activations of features labeled as *anomaly* by a factor $\alpha \geq 1$, while leaving the remaining features unchanged. The joint intervention applies both operations simultaneously. Thus, for the joint intervention,

$$\tilde{h}_{p,j} = \begin{cases} 0, & j \in \mathcal{J}_{\mathrm{dist}}, \\ \alpha h_{p,j}, & j \in \mathcal{J}_{\mathrm{anom}}, \\ h_{p,j}, & \text{otherwise}, \end{cases} \tag{5}$$

where $\mathcal{J}_{\mathrm{dist}}$ and $\mathcal{J}_{\mathrm{anom}}$ denote the sets of features labeled as distractors and anomalies, respectively. Features labeled as *uncertain* are therefore left unchanged. These interventions are applied at every patch where the corresponding feature is active.

Because the SAE does not perfectly reconstruct the original residual, we preserve its reconstruction error during the intervention. For each patch, we define:

$$\boldsymbol{\epsilon}_p = \mathbf{r}_p - \hat{\mathbf{r}}_p \tag{6}$$

and reconstruct the intervened residual and patch embedding as:

$$\tilde{\mathbf{r}}_p = \mathbf{W}\tilde{\mathbf{h}}_p + \boldsymbol{\epsilon}_p, \qquad \tilde{\mathbf{z}}_p = \mathbf{m}_p + \tilde{\mathbf{r}}_p \tag{7}$$

so that the intervention changes only the contribution represented by the SAE features while leaving the reconstruction error unchanged. This preserves residual information not captured by the SAE, including information that may be relevant to anomalies not represented in the calibration set.

For interventions involving amplification, we select a single category-level factor α from the fixed set $\{1, 1.5, 2, 3, 4, 5\}$ that maximizes image-level AUROC on the source calibration set. When multiple values achieve the same maximum AUROC, we select the smallest α. We compare this selection procedure with fixed amplification factors in Appendix C. The selected value is then fixed for both source and corruption evaluations. After reconstructing the edited patch embeddings, we perform a fresh nearest-neighbor search over the unchanged normal memory bank $\mathcal{M}$:

$$\tilde{\mathbf{m}}_p = \underset{\mathbf{m}\in\mathcal{M}}{\arg\min} \ \|\tilde{\mathbf{z}}_p - \mathbf{m}\|_2 \tag{8}$$

The resulting nearest-neighbor distances and matches are then passed through the same frozen PatchCore scoring rule to obtain the intervened anomaly scores.

4 Experimental Setup

Datasets. We evaluate on VisA (Zou et al., 2022), MVTec AD (Bergmann et al., 2019), MVTec LOCO AD (Bergmann et al., 2022), and MVTec AD 2 (Heckler-Kram et al., 2026), covering 40 categories in total, and separately on RobustAD (Pemula et al., 2025) under real acquisition shifts. For each category in the first four benchmarks, we construct a balanced calibration set, with the number of images per class corresponding to 20% of the smaller test class, and use the remaining images for evaluation. The resulting calibration sets contain a median of 20 images per category. Because RobustAD already includes anomalous images in its training split, we instead construct the calibration set directly from the training data, using all anomalous images and an equal number of normal images, with a median of 150 images per category. For MVTec AD 2, where the same physical scene is acquired under multiple conditions, the same sampling rule is applied to scene groups, keeping all acquisitions of the same scene together to avoid closely related views crossing the calibration and test splits. For MVTec LOCO AD and RobustAD, which include descriptions of the expected normal configuration, we additionally provide these descriptions to the MLLM.

Implementation Details. We use PatchCore with a frozen DINOv3 ViT-L/16 backbone, extracting representations from `blocks.11` and `blocks.23` at an input resolution of 512×512, with the default coreset ratio of 0.1 and 9 neighbors for final image scoring. For each category, we train a separate SAE with 64 features and TopK $k = 4$. SAE-specific hyperparameters and metrics, together with alternative widths and sparsity levels, are reported in Appendix B, and alternative vision backbones in Appendix F. For the automated interpretability step, we use GPT-5.6 Sol with reasoning effort set to `none` and temperature 0. At the time of writing the paper, a complete 64-feature annotation pass costs approximately USD 3. We evaluate open-weight MLLMs in Appendix D, and provide an example annotation prompt in Appendix H.

Evaluation Protocol. We report image-level AUROC and macro-average results, giving each category equal weight. Results on VisA, MVTec AD, MVTec LOCO AD, and MVTec AD 2 are macro-averaged across their 40 categories, while RobustAD is reported separately across its three categories. We compare the original PatchCore with two direct uses of the available calibration data and with three strategies for assigning SAE features to interventions. First, we evaluate PatchCore with the normal calibration images included in memory-bank construction while retaining the original coreset ratio of 0.1. We also evaluate a class-weighted DINOv3 linear probe on globally average-pooled patch representations from the same frozen backbone. The probe is trained on known-normal training images and both normal and anomalous calibration images. The purpose

Table 1: Image-level AUROC on source and synthetically corrupted data, macro-averaged across categories. Best results are in bold. Random interventions are averaged over 10 seeds.

Method	Intervention	Evaluation	VisA (12)	MVTec AD (15)	MVTec LOCO (5)	MVTec AD 2 (8)	All (40)
Original PatchCore		Source	0.8953	0.9859	0.8430	0.6440	0.8724
		Corrupted	0.7868	0.9311	0.7831	0.6177	0.8066
PatchCore + extra normals		Source	0.8955	0.9866	0.8482	0.6521	0.8751
		Corrupted	0.7862	0.9316	0.7874	0.6236	0.8083
DINOv3 linear probe		Source	0.8979	0.8544	0.6199	0.5998	0.7872
		Corrupted	**0.8664**	0.8296	0.6076	0.6094	0.7688
Random	Suppression	Source	0.8903	0.9836	0.8390	0.6542	0.8716
		Corrupted	0.7722	0.9191	0.7814	0.6233	0.7987
	Amplification	Source	0.8593	0.9832	0.8397	0.6249	0.8564
		Corrupted	0.7632	0.9272	0.7856	0.6036	0.7956
	Joint	Source	0.8581	0.9761	0.8393	0.6304	0.8545
		Corrupted	0.7556	0.9128	0.7814	0.6100	0.7887
Calibration-guided	Suppression	Source	0.9006	0.9886	0.8490	0.6581	0.8787
		Corrupted	0.7928	0.9358	0.7938	0.6271	0.8134
	Amplification	Source	0.9052	0.9874	0.8554	0.6478	0.8783
		Corrupted	0.8057	0.9328	**0.8000**	0.6162	0.8148
	Joint	Source	0.9061	0.9882	**0.8570**	0.6579	0.8811
		Corrupted	0.8080	0.9371	0.7986	0.6269	0.8190
AutoInterp-guided	Suppression	Source	0.9080	**0.9893**	0.8419	0.6612	0.8808
		Corrupted	0.7923	0.9352	0.7891	**0.6276**	0.8126
	Amplification	Source	0.9198	0.9849	0.8400	0.6580	0.8819
		Corrupted	0.8134	0.9312	0.7867	0.6184	0.8152
	Joint	Source	**0.9242**	0.9886	0.8419	**0.6626**	**0.8857**
		Corrupted	0.8162	**0.9389**	0.7891	0.6269	**0.8210**

of these baselines is to test the effect of using the available calibration data directly without any sparse feature decomposition or automated interpretability. For feature interventions, we compare AutoInterp-guided, calibration-guided, and random feature assignments, applying suppression, amplification, and joint intervention under each strategy. For the calibration-guided assignment, we compute the calibration AUROC of each SAE feature, treating features below 0.5 as distractors and those above 0.5 as anomaly features, with ties left unchanged. This provides an alternative protocol testing whether the AutoInterp procedure adds value beyond assigning feature labels directly from calibration supervision. For the random assignment, we sample feature sets matching the numbers of AutoInterp-labeled distractor and anomaly features for each category. The two sets are disjoint for the joint intervention. Random amplification and joint intervention use the corresponding category-level amplification factor selected by the AutoInterp procedure. This control tests whether performance depends on which SAE features are selected for suppression and amplification, while keeping the number of selected features and the corresponding amplification multiplier fixed.

To evaluate robustness to synthetic image corruptions, we apply brightness, contrast, defocus blur, and Gaussian noise from `imagecorruptions` at severities 1–5 (Hendrycks & Dietterich, 2019; Michaelis et al., 2019). We additionally evaluate robustness to real acquisition shifts on RobustAD. In both evaluations, feature assignments and amplification factors are determined using only source calibration data and remain fixed across the corrupted or shifted evaluation data.

5 Results and Discussion

RQ1: Performance on source data. Table 1 reports performance across the 40 categories of VisA, MVTec AD, MVTec LOCO AD, and MVTec AD 2. On source data, AutoInterp-guided interventions improve the macro-average AUROC of the original PatchCore from 0.8724 to 0.8808

Table 2: Image-level AUROC on RobustAD source and acquisition shifts, macro-averaged across categories. Best results are in bold. Random interventions are averaged over 10 seeds.

Method	Intervention	Evaluation	PCB	MetalParts	PiledBags	All (3)
Original PatchCore		Source	0.8926	0.7423	0.9887	0.8745
		Shifted	0.3943	0.6415	0.7849	0.6069
PatchCore + extra normals		Source	0.9202	0.7386	0.9889	0.8826
		Shifted	0.4154	0.6413	0.7828	0.6131
DINOv3 linear probe		Source	**0.9998**	0.7495	0.8057	0.8517
		Shifted	0.5165	**0.7106**	0.7177	0.6482
Random	Suppression	Source	0.8587	0.7572	0.9652	0.8604
		Shifted	0.4271	0.6363	0.7424	0.6019
	Amplification	Source	0.9509	0.7193	0.9774	0.8825
		Shifted	0.4368	0.6402	0.7994	0.6254
	Joint	Source	0.9495	0.7573	0.9651	0.8906
		Shifted	0.4401	0.6363	0.7423	0.6062
Calibration-guided	Suppression	Source	0.8946	0.7479	**0.9993**	0.8806
		Shifted	0.3956	0.6424	0.7769	0.6049
	Amplification	Source	0.9722	0.6587	0.9990	0.8766
		Shifted	0.2928	0.6269	0.8247	0.5815
	Joint	Source	0.9722	0.7015	**0.9993**	0.8910
		Shifted	0.2891	0.6255	0.7769	0.5638
AutoInterp-guided	Suppression	Source	0.8822	**0.7747**	0.9958	0.8842
		Shifted	0.4899	0.6574	0.8016	0.6496
	Amplification	Source	0.9456	0.7374	0.9921	0.8917
		Shifted	0.5149	0.6568	**0.8452**	**0.6723**
	Joint	Source	0.9464	**0.7747**	0.9958	**0.9056**
		Shifted	**0.5208**	0.6574	0.8016	0.6599

with distractor suppression, 0.8819 with anomaly amplification, and 0.8857 with their joint intervention. The joint intervention improves performance on three of the four dataset families, with MVTec LOCO AD being the exception. Improvements are also observed when the original detector is already close to saturation. For example, on MVTec AD, PatchCore reaches 0.9859 AUROC, while AutoInterp-guided suppression reaches 0.9893.

The controls suggest that these gains are not explained by simply using the calibration data. Adding normal calibration images to PatchCore memory bank improves overall AUROC by only 0.27 points. A linear probe trained on all normal training images plus the calibration set performs substantially worse, reaching 0.7872 AUROC. This may partly reflect the limited amount of supervised data available, with a median of only 10 anomalous calibration examples per category. Calibration-guided interventions make more effective use of the same calibration supervision, with joint intervention reaching 0.8811 and outperforming AutoInterp on MVTec LOCO AD, but remaining below AutoInterp overall. Among the intervention procedures, as expected, random feature assignment gives the weakest results overall, but notably does not collapse the detector. We initially hypothesized that this tolerance to arbitrary edits might arise from the substantial signal retained in the reconstruction error alone. However, this tolerance persists when the reconstruction error is omitted (see Appendix B).

RQ2: Robustness to corruptions and acquisition shifts. We next test whether the improvements obtained on source data persist when the input distribution changes. Under synthetic corruptions, the joint AutoInterp-guided intervention improves macro-average AUROC from 0.8066 to 0.8210, a gain of 1.44 points, slightly larger than the 1.33-point gain observed on source data. The improvement persists at every corruption severity, with the largest gap at severity 5. At this severity, joint intervention improves AUROC by 1.83 points on average and outperforms the original PatchCore in 30 of the 40 categories. On RobustAD, the improvement is particularly large under real acquisition shifts. Joint intervention improves source AUROC from 0.8745 to 0.9056, a gain of 3.11 points,

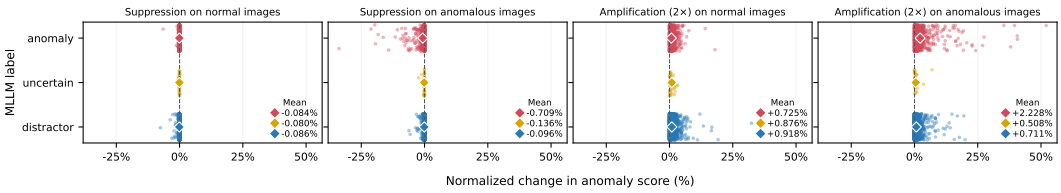


Figure 2: Effects of suppressing and amplifying SAE features across 43 categories. Points show normalized mean score changes per feature, and diamonds show category-balanced means by label.

while under shifted acquisition conditions it improves AUROC from 0.6069 to 0.6599, a gain of 5.30 points. The individual suppression and amplification interventions show the same qualitative pattern, with larger gains under shifted acquisition conditions than on source data. RobustAD also provides substantially more calibration data than the other benchmarks, which could contribute to better SAE estimation. Its calibration sets contain a median of 150 images per category, compared with a median of 20 images per category across the 40-category benchmark. However, within those 40 categories, whose calibration sets range from 4 to 54 images, calibration-set size shows only a modest positive association with joint-intervention gain on source data (Spearman's $\rho = 0.371$), which becomes substantially weaker under corruptions ($\rho = 0.191$).

Regarding the calibration-guided control, the joint intervention is competitive under synthetic corruptions, reaching 0.8190 AUROC compared with 0.8210 for AutoInterp. Under the real acquisition shifts of RobustAD, however, it reaches 0.5638, compared with 0.6069 for the original PatchCore and 0.6599 for AutoInterp. Since AutoInterp and calibration guidance use the same calibration data and SAE, their different behavior suggests that semantic feature interpretation provides information beyond calibration AUROC, especially under real acquisition shifts.

The linear probe shows a heterogeneous pattern across both robustness evaluations. Under synthetic corruptions, it substantially outperforms the original PatchCore on VisA but falls below it on MVTec AD, MVTec LOCO AD, and MVTec AD 2. Under the real acquisition shifts of RobustAD, it outperforms PatchCore on PCB and MetalParts but falls below it on PiledBags. This heterogeneous behavior suggests that direct prediction from the calibration data may be highly sensitive to how well the available examples represent the evaluation distribution.

RQ3: Analysis of individual feature interventions by MLLM label. We next test whether the MLLM labels predict the class-specific effects of individual SAE feature interventions on PatchCore scores. For each non-dead feature, we independently suppress or amplify it across all patches of every test image, rerun PatchCore, and measure the resulting change in image anomaly score separately for normal and anomalous images. Score changes are normalized by the category's mean original test score, computed over all test images. We use a fixed $2\times$ amplification factor, close to the mean automatically selected amplification multiplier of $1.884\times$ across the 43 categories. Of the 64 SAE features per category, an average of 63.86 are non-dead, with 22.44 labeled as anomaly, 40.58 as distractor, and 0.84 as uncertain. In Figure 2, each point represents the average score change produced by one feature across all images of the corresponding class. Consistent with what we would expect from anomaly and distractor features, suppressing anomaly features reduces scores more on anomalous than on normal images, while amplifying them increases scores more on anomalous images. In contrast, distractor features have similar effects on the two classes.

To assess whether these class-specific differences are consistent across the 43 categories, we perform four statistical tests, one for each combination of feature label and intervention. For each test, we compare category-level average score changes on anomalous images with corresponding averages on normal images. For anomaly features, we use one-sided paired t-tests in the predicted direction. For distractor features, we use two-sided paired t-tests because no directional class difference is predicted. We apply Holm correction across the four tests. For anomaly features, the class-specific difference is significant under both suppression ($p_{\mathrm{Holm}} = 2.07 \times 10^{-6}$) and amplification ($p_{\mathrm{Holm}} = 1.73 \times 10^{-6}$), supporting the pattern observed in Figure 2. For distractor features, the difference is not significant under either suppression ($p_{\mathrm{Holm}} = 0.346$) or amplification ($p_{\mathrm{Holm}} = 0.0755$), with distractor amplification showing a larger increase on normal images. Uncertain features are excluded from these tests because they are few and occur in only 15 of 43 categories.

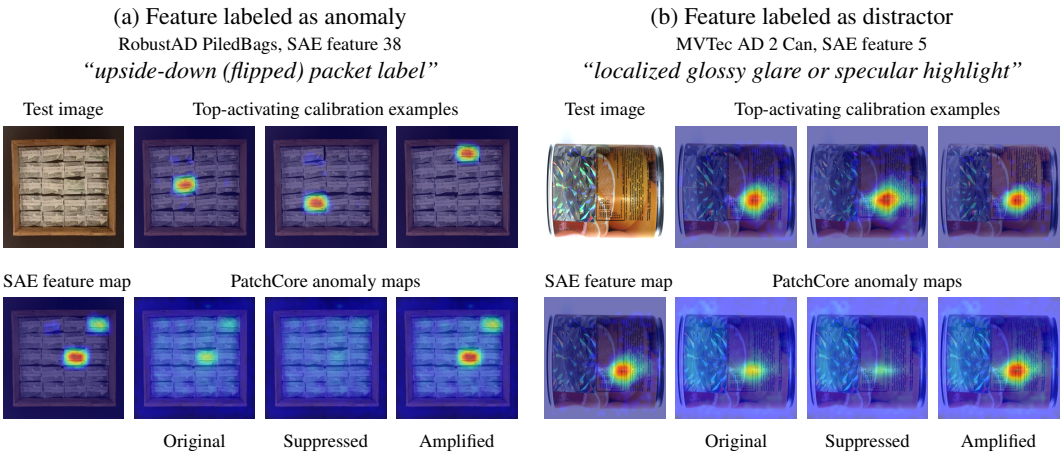


Figure 3: Examples of anomaly and distractor SAE features. Italic text gives the MLLM-generated feature description. The top row shows a test image and three top-activating calibration examples shown to the MLLM. The bottom row shows, for the test image, the SAE feature map and PatchCore anomaly maps before intervention, after suppression, and after amplification. The three PatchCore anomaly maps for each example share a min-max color scale.

Another pattern visible in Figure 2 is that the separation between features labeled as anomaly and features labeled as distractor is concentrated toward the tails of the distributions. At the 10th percentile for suppression and the 90th percentile for amplification, the difference between anomaly and distractor score changes is 1.71 and 2.03 times the corresponding average difference across all features, respectively. Figure 3 provides qualitative examples of one feature labeled as anomaly and one labeled as distractor, together with their effects on PatchCore anomaly maps. In the shown examples, suppressing the PiledBags feature described as *"upside-down (flipped) packet label"* reduces the score of the anomalous image by approximately 42.3%, while suppressing the Can feature described as *"localized glossy glare or specular highlight"* reduces the score of the normal image by 29.32%. These examples show much larger score changes than the average effects in Figure 2, as the latter averages the effect over all images of the corresponding class, including many where the feature has little or no effect. More qualitative examples are provided in Appendix G.

6 Conclusion and Limitations

We presented an approach for augmenting visual anomaly detectors by decomposing their anomaly signal into sparse features, automatically interpreting these features with an MLLM, and using the resulting labels to guide interventions on the detector. Across a total of 43 categories from five benchmarks, these interventions improve PatchCore performance on source data and robustness under distribution shift. The gains persist under synthetic corruptions and are particularly large under the real acquisition shifts of RobustAD. At the feature level, we further find that the MLLM labels are aligned in aggregate with how the corresponding features affect normal and anomalous images. Together, these results show that automatically interpreted sparse features can provide an actionable interface to the anomaly signal of a frozen detector.

Our study also has important limitations. First, the method requires a calibration set containing anomalous examples, and the amplification factor is selected using calibration AUROC. The intervention procedure is therefore not purely normal-only, even when applied to an unsupervised anomaly detector. Second, although the interventions improve performance on average, they do not guarantee an improvement for every category or model configuration and can degrade the original detector in some cases. Finally, the generated feature descriptions and labels should be viewed as hypotheses about feature semantics, since the MLLM may misinterpret the provided examples and individual SAE features may themselves remain polysemantic. Our feature-level analysis shows that their behavior is consistent with the assigned labels in aggregate, but there is no ground-truth annotation for individual SAE features and therefore no guarantee that each interpretation is faithful.

AI USE STATEMENT

In this work, we used generative AI tools to propose or refine hypotheses, provide feedback on research methodology and experiments, implement methods, and interpret results. We have not used generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, or clean and reformat datasets. Formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translation, and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to create or edit software code, draft parts of the research paper, brainstorm, and edit the research paper to improve readability. We have reviewed all AI-assisted work. LLM-generated code was reviewed and tested for correctness, and LLM-generated manuscript content was heavily edited by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Kilian Batzner, Lars Heckler, and Rebecca König. Efficientad: Accurate visual anomaly detection at millisecond-level latencies. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 128–138, January 2024.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.

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 (CVPR)*, June 2019.

Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. *Int. J. Comput. Vision*, 130(4):947–969, April 2022. ISSN 0920-5691. doi: 10.1007/s11263-022-01578-9. URL https://doi.org/10.1007/s11263-022-01578-9.

Matteo Bianchi, Riccardo Campi, Antonio De Santis, Sara Merengo, and Marco Brambilla. Activation-based concept extraction for explainability in image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 3996–4005, June 2026.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html, 2023.

Trenton Bricken, Adly Templeton, Ben Chen, Jack Lindsey, Adam Jermyn, Shan Carter, Tom Henighan, Adam Pearce, and Chris Olah. Towards monosemanticity: Decomposing language models with dictionary learning. https://transformer-circuits.pub/2023/monosemantic-features, 2023. Transformer Circuits Thread, Anthropic.

Antonio De Santis, Riccardo Campi, Matteo Bianchi, and Marco Brambilla. Visual-TCAV: Concept-based attribution and saliency maps for post-hoc explainability in image classification. *Transactions on Machine Learning Research*, 2026a. ISSN 2835-8856. URL https://openreview.net/forum?id=SLh00W5rhu.

Antonio De Santis, Schrasing Tong, Marco Brambilla, and Lalana Kagal. Learning concept bottleneck models from mechanistic explanations. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=gdEWoxhb70.

Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: A patch distribution modeling framework for anomaly detection and localization. In *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021,*

Proceedings, Part IV, pp. 475–489, Berlin, Heidelberg, 2021. Springer-Verlag. ISBN 978-3-030-68798-4. doi: 10.1007/978-3-030-68799-1_35. URL https://doi.org/10.1007/978-3-030-68799-1_35.

Hanqiu Deng and Xingyu Li. Anomaly detection via reverse distillation from one-class embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9737–9746, June 2022.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022. URL https://arxiv.org/abs/2209.10652.

Thomas Fel, Agustin Picard, Louis Béthune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2711–2721, June 2023.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tcsZt9ZNKD.

Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

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 (WACV)*, pp. 98–107, January 2022.

Lars Heckler-Kram, Jan-Hendrik Neudeck, Ulla Scheler, Rebecca König, and Carsten Steger. The mvtec ad 2 dataset: Advanced scenarios for unsupervised anomaly detection. *International Journal of Computer Vision*, 134:175, 2026. doi: 10.1007/s11263-026-02743-0.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Rui Jiang, Yijia Xue, and Dongmian Zou. Interpretability-aware industrial anomaly detection using autoencoders. *IEEE Access*, 11:60490–60500, 2023. doi: 10.1109/ACCESS.2023.3286548.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2668–2677. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kim18d.html.

Philipp Liznerski, Lukas Ruff, Robert A. Vandermeulen, Billy Joe Franks, Marius Kloft, and Klaus Robert Muller. Explainable deep one-class classification. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=A5VV3UyIQz.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=I4e82CIDxv.

Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.

Ibrahima Ndiour, Nilesh A Ahuja, Ergin U Genc, and Omesh Tickoo. Fre: A fast method for anomaly detection and segmentation. In *34th British Machine Vision Conference 2023, BMVC 2023, Aberdeen, UK, November 20-24, 2023*. BMVA, 2023. URL https://papers.bmvc2023.org/0614.pdf.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Robust ad: A real world benchmark dataset for robustness in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 4086–4096, June 2025.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.828. URL https://aclanthology.org/2024.acl-long.828/.

Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14298–14308, 2022. doi: 10.1109/CVPR52688.2022.01392.

Tamar Rott Shaham, Sarah Schwettmann, Franklin Wang, Achyuta Rajaram, Evan Hernandez, Jacob Andreas, and Antonio Torralba. A multimodal automated interpretability agent. In *Forty-first International Conference on Machine Learning*, 2024.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Pei Wang, Wei Zhai, and Yang Cao. Robustness benchmark for unsupervised anomaly detection models. *JUSTC*, 54(1):0103–1–0103–11, 2024. ISSN 2097-7387. doi: 10.52396/JUSTC-2022-0165. URL https://justc.ustc.edu.cn/en/article/doi/10.52396/JUSTC-2022-0165.

Yizhou Wang, Dongliang Guo, Sheng Li, Octavia Camps, and Yun Fu. Unveiling the unseen: A comprehensive survey on explainable anomaly detection in images and videos, 2025. URL https://arxiv.org/abs/2302.06670.

Ruihan Zhang, Prashan Madumal, Tim Miller, Krista A. Ehinger, and Benjamin I. P. Rubinstein. Invertible concept-based explanations for cnn models with non-negative concept activation vectors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(13):11682–11690, May 2021. doi: 10.1609/aaai.v35i13.17389. URL https://ojs.aaai.org/index.php/AAAI/article/view/17389.

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

A Appendix Overview

In the appendix, we provide:

B. SAE training, metrics, and ablations

C. Amplification multiplier selection
D. Multimodal LLM ablation
E. Anomaly detector ablation
F. Vision backbone ablation
G. Additional qualitative examples
H. Example MLLM annotation prompt

B SAE TRAINING, METRICS, AND ABLATIONS

Training and reconstruction fidelity. We train one SAE per category on the PatchCore residuals extracted from the calibration images. Each SAE has width $m = 64$ and TopK sparsity $k = 4$, with tied encoder and decoder weights, no biases, and unit-norm feature directions. We optimize reconstruction MSE with Adam using a learning rate of 10^{-4} and batch size 2048 for up to 1000 epochs, with early stopping based on training reconstruction MSE (patience 50) and restoring the checkpoint with the lowest training MSE. Table 3 reports training and test reconstruction MSE and cosine similarity, the number of dead features, and recovered AUROC. Recovered AUROC is the AUROC obtained using the residual reconstructed by the SAE without adding the reconstruction error, divided by the original PatchCore AUROC and expressed as a percentage. The reconstructed descriptors are rematched to the original PatchCore memory bank and rescored with the unchanged detector. Across the 40 RQ1 categories, the SAE reconstruction recovers 95.95% of the original PatchCore AUROC on average, while almost all features remain active. The corresponding recovered AUROC on RobustAD is 98.74%.

Table 3: SAE reconstruction fidelity evaluated on the test split, macro-averaged across categories. Recovered AUROC is reported as a percentage of the original PatchCore AUROC.

Metric	VisA (12)	MVTec AD (15)	MVTec LOCO (5)	MVTec AD 2 (8)	All (40)
Training MSE	21.11	30.41	26.86	42.86	29.67
Test MSE	25.00	52.09	31.80	57.43	42.50
Training cosine	0.550	0.633	0.568	0.646	0.602
Test cosine	0.504	0.513	0.550	0.500	0.512
Recovered AUROC (%)	97.23	97.55	92.92	91.15	95.95
Dead features / 64	0.00	0.00	0.00	0.25	0.05

Table 4: SAE reconstruction fidelity evaluated on the RobustAD source test split. Recovered AUROC is reported as a percentage of the original PatchCore AUROC.

Metric	PCB	MetalParts	PiledBags	All (3)
Training MSE	13.55	32.28	28.36	24.73
Test MSE	13.49	32.82	33.10	26.47
Training cosine	0.599	0.553	0.679	0.610
Test cosine	0.586	0.538	0.645	0.590
Recovered AUROC (%)	110.37	84.72	98.74	98.74
Dead features / 64	0.00	0.00	4.00	1.33

Tied and untied SAE ablation. We compare the tied, bias-free SAE used in the main experiments with a more canonical untied variant that uses separate encoder and decoder weights and learned biases, while keeping the same calibration residuals, width $m = 64$, TopK sparsity $k = 4$, optimizer, and maximum number of epochs. Across the 40 RQ1 categories, the two architectures obtain similar reconstruction MSE and recovered AUROC, while the tied SAE has substantially fewer dead features on average, 0.05 compared with 11.43 out of 64. Even without the tying constraint, the

untied SAE still tends to learn features that contribute positively to the PatchCore anomaly signal, with only 3.34% of individual feature removals increasing the raw nearest-neighbor patch distance. At comparable reconstruction and recovered AUROC, we therefore retain the tied SAE for our intervention setting, since individual feature removal is guaranteed not to increase this distance, with 0% observed violations.

Table 5: Comparison of tied and untied SAEs across the 40 RQ1 categories.

Metric	Tied, bias-free	Untied, with biases
Test reconstruction MSE	42.50	43.35
Recovered AUROC (%)	95.95	95.82
Dead features / 64	0.05	11.43
Removals increasing patch distance (%)	0.00	3.34

Width and sparsity sensitivity. We evaluate the complete pipeline for dictionary widths $m \in \{32, 64, 128\}$ and TopK sparsities $k \in \{2, 4, 8\}$ on two categories, VisA *capsules* and MVTec AD *pill*. We restrict this ablation to two categories because there are 9 configurations per category and each requires training a new SAE, generating new feature labels through MLLM calls, and rerunning the downstream interventions. For each of the 9 configurations, we evaluate suppression, amplification, and joint intervention on source and corrupted data using the same procedure as in the main experiments. Suppression and joint intervention improve over the original detector for both categories across all tested configurations, although the magnitude of the improvement varies significantly and does not change monotonically with either dictionary width or TopK sparsity. These results suggest that there may not be a single overall best configuration among the nine tested settings, although experiments on more categories would be needed to establish this more clearly.

Table 6: Width and TopK sensitivity on VisA *capsules* and MVTec AD *pill*, macro-averaged across the two categories.

Width m	TopK k	Source			Corrupted		
		Suppression	Amplification	Joint	Suppression	Amplification	Joint
Original PatchCore		0.8592			0.7836		
32	2	0.8816	**0.9418**	**0.9465**	0.7998	0.8544	0.8571
32	4	0.8924	0.9263	0.9373	0.8027	0.8545	0.8543
32	8	0.8903	0.9123	0.8903	0.8023	0.8414	0.8023
64	2	0.8858	0.8821	0.8965	0.7969	0.8097	0.8126
64	**4**	0.8890	0.9245	0.9265	0.7951	**0.8545**	**0.8580**
64	8	0.8975	0.8834	0.8975	0.8021	0.8001	0.8021
128	2	0.8778	0.8830	0.8876	0.7954	0.8037	0.7990
128	4	0.8846	0.8863	0.8739	0.7990	0.7964	0.7885
128	8	**0.9071**	0.9116	0.9071	**0.8067**	0.8164	0.8067

Dependence on reconstruction error. Our main intervention preserves the SAE reconstruction error $\boldsymbol{\epsilon}_p = \mathbf{r}_p - \hat{\mathbf{r}}_p$ by design. Because the SAE is trained only on calibration residuals, replacing the original residual with its sparse reconstruction would discard information not captured by the dictionary learned from the calibration set. This would be undesirable in anomaly detection, where held-out anomalies need not resemble those available during calibration. Preserving $\boldsymbol{\epsilon}_p$ therefore keeps the residual information outside the SAE representation while allowing the intervention to edit only the decoded sparse contributions.

We nevertheless omit $\boldsymbol{\epsilon}_p$ as a diagnostic to measure how the SAE reconstruction performs on its own and whether the intervention gains remain when evaluation is restricted to this representation. Amplification and joint intervention select their multiplier independently with and without reconstruction error using the same clean-calibration procedure as in the main experiments.

Without reconstruction error, the unedited SAE representation reaches 0.8368 source AUROC, substantially below the original PatchCore value of 0.8724. A complementary source-only diagnostic

Table 7: Effect of retaining the SAE reconstruction error across the 40 RQ1 categories, macro-averaged across categories. For no intervention, the error-retained condition corresponds to original PatchCore, while the error-omitted condition corresponds to the SAE-only reconstruction.

Intervention	Source		Corrupted	
	Error retained	Error omitted	Error retained	Error omitted
None	0.8724	0.8368	0.8066	0.7760
Suppression	0.8808	0.8516	0.8126	0.8102
Amplification	0.8819	0.8524	0.8152	0.7964
Joint	**0.8857**	**0.8551**	**0.8210**	**0.8136**

shows the opposite component: retaining only the reconstruction error yields 0.8697 AUROC, close to the original detector. These component evaluations are not an additive decomposition of AUROC because memory-bank rematching and image-level scoring are nonlinear.

Importantly, the intervention gains remain visible when evaluation is restricted to the SAE reconstruction. Relative to the no-intervention SAE-only baseline, suppression, amplification, and joint intervention increase source AUROC from 0.8368 to 0.8516, 0.8524, and 0.8551, respectively. Under corruption, they increase AUROC from 0.7760 to 0.8102, 0.7964, and 0.8136. Thus, the reconstruction error remains important for preserving the detector information outside the sparse representation, while the benefits of the interventions are also present within the SAE representation itself.

We additionally test random feature interventions without reconstruction error on source and corrupted data. The results are shown in Table 8.

Table 8: AUROC for random interventions without SAE reconstruction error across the 40 RQ1 categories, macro-averaged across categories. None corresponds to the unedited SAE-only reconstruction. Random interventions use a single feature assignment.

Intervention	Source	Corrupted
None	**0.8368**	**0.7760**
Suppression	0.7807	0.7274
Amplification	0.8273	0.7715
Joint	0.7899	0.7348

Random interventions reduce performance relative to the unedited SAE-only representation, but do not collapse the detector even when the reconstruction error is absent. The same pattern holds under corruptions. On source data, AutoInterp remains higher under all three interventions, showing that its gains within the SAE representation depend on which features are edited rather than on sparse intervention alone.

C AMPLIFICATION MULTIPLIER SELECTION

We evaluate how the amplification multiplier affects performance for both amplification-only and joint intervention. We vary $\alpha \in \{1, 1.5, 2, 3, 4, 5\}$ across the 40 RQ1 categories. In the amplification-only setting, anomaly features are multiplied by α, while distractor and uncertain features are left unchanged. In the joint setting, distractor features are additionally suppressed. Thus, $\alpha = 1$ leaves the original detector unchanged for amplification-only and reduces the joint intervention to distractor suppression. We also report the setting used in the main experiments, where α is selected independently for each category using clean calibration AUROC and then kept fixed during source and corrupted evaluation.

For joint intervention, performance increases for moderate amplification strengths and decreases on source data for larger multipliers. Among the fixed values, $2\times$ obtains the highest source AUROC and $3\times$ the highest corrupted AUROC. Amplification-only shows a similar sensitivity to the

Table 9: Effect of the anomaly-feature amplification multiplier on amplification-only and joint intervention across the 40 RQ1 categories, macro-averaged across categories. Best results in each column are in bold.

	Source		Corrupted	
Multiplier	Amplification	Joint	Amplification	Joint
$1\times$	0.8724	0.8808	0.8066	0.8126
$1.5\times$	0.8761	0.8831	0.8155	0.8199
$2\times$	0.8785	0.8836	0.8210	0.8245
$3\times$	0.8801	0.8789	**0.8246**	**0.8259**
$4\times$	0.8745	0.8735	0.8236	0.8237
$5\times$	0.8703	0.8702	0.8210	0.8212
Calibration-selected	**0.8819**	**0.8857**	0.8152	0.8210

multiplier, with performance increasing up to $3\times$ on both source and corrupted data before decreasing at larger values. The calibration-selected setting obtains the highest source AUROC for both interventions, but does not maximize corrupted AUROC.

These results also highlight a limitation of the current multiplier selection strategy. Selecting α independently on a small clean calibration set may overfit category-specific variation, while using a single fixed multiplier across all categories does not account for differences in intervention sensitivity. More robust strategies for selecting intervention strength may therefore be worth exploring.

D Multimodal LLM ablation

We evaluate the sensitivity of the method to the MLLM used for feature interpretation on a fixed subset of ten categories. We compare GPT-5.6 Sol, used in the main experiments, with the open-weight Gemma 4 31B-IT and Qwen3.5-27B. The detector, SAE, interpretation protocol, and downstream intervention procedure are kept unchanged. For each annotator, we evaluate suppression, amplification-only, and joint intervention on source and corrupted data.

Table 10: Effect of the MLLM used for feature interpretation on a fixed subset of ten categories, macro-averaged across categories.

	Source			Corrupted		
MLLM	Suppression	Amplification	Joint	Suppression	Amplification	Joint
Original PatchCore	0.8653			0.7951		
GPT-5.6 Sol	0.8779	0.8808	0.8849	0.8024	0.8157	**0.8243**
Gemma 4 31B-IT	**0.8786**	**0.8853**	**0.8899**	0.8006	**0.8165**	0.8207
Qwen3.5-27B	0.8773	0.8708	0.8791	**0.8043**	0.8055	0.8149

Joint intervention improves over the original detector on both source and corrupted data for all three MLLMs. The exact semantic labels are more model-dependent, with pairwise agreement of 81.1% between GPT and Gemma, 69.5% between GPT and Qwen, and 67.8% between Gemma and Qwen. Despite these differences, the downstream intervention remains beneficial across all three annotators, suggesting that the observed effect is not specific to the MLLM used in the main experiments.

E Anomaly detector ablation

To test whether our approach extends beyond nearest-neighbor anomaly detection, we repeat the pipeline with FRE (Ndiour et al., 2023), a feature-reconstruction detector, on five categories: Capsules, Pill, Grid, Transistor, and Pushpins. We use the native Anomalib FRE implementation with a frozen DINOv3 ViT-L/16 backbone, extracting representations from `blocks.23` at an input resolution of 256×256. FRE applies 2×2 average pooling and flattens the resulting $1024 \times 8 \times 8$

feature tensor before reconstructing it with a tied linear autoencoder with a 220-dimensional latent representation. The FRE autoencoder is trained with mean squared reconstruction error and Adam with learning rate 10^{-3}, batch size 128, and a fixed 440 epochs.

The resulting FRE reconstruction residual is decomposed using a separate tied TopK SAE with 64 features and $k = 4$ for each category. Automated interpretability and interventions follow the same protocol as in the main experiments. Distractor features are suppressed, anomaly features are amplified, and uncertain features are left unchanged. Amplification-only and joint intervention select their multiplier from $\{1, 1.5, 2, 3, 4, 5\}$ using source calibration AUROC and keep it fixed under corruption. PatchCore uses its main configuration on the same five categories.

Table 11: Anomaly detector ablation on five categories. Deltas are AUROC percentage points relative to the corresponding original detector. Best intervention results for each detector and evaluation setting are in bold.

Detector	Intervention	Source		Corrupted	
		AUROC	Δ	AUROC	Δ
PatchCore	Original	0.8969	–	0.8428	–
	Suppression	0.9105	+1.36	0.8464	+0.37
	Amplification	0.9166	+1.96	0.8714	+2.86
	Joint	**0.9255**	**+2.86**	**0.8716**	**+2.88**
FRE	Original	0.8475	–	0.7488	–
	Suppression	**0.8679**	**+2.03**	0.7462	-0.26
	Amplification	0.8501	+0.26	**0.7602**	**+1.14**
	Joint	0.8675	+2.00	0.7479	-0.09

With FRE, suppression and joint intervention improve source AUROC by 2.03 and 2.00 points, respectively. Under synthetic corruptions, however, their average effects are close to zero, with changes of -0.26 and -0.09 points. Amplification shows the opposite pattern, producing only a small source gain of 0.26 points but improving corrupted AUROC by 1.14 points. The corresponding PatchCore interventions produce positive average gains in both evaluations on the same five categories. These results provide some preliminary evidence that the intervention framework could extend to a feature-reconstruction detector. However, experiments on more categories would be needed to draw stronger conclusions.

F VISION BACKBONE ABLATION

To test whether the effect of our interventions transfers beyond the DINOv3 backbone used in the main experiments, we repeat the complete pipeline with SwinV2-L and WideResNet-50-2 on the same five categories used in the detector ablation, Capsules, Pill, Grid, Transistor, and Pushpins. SwinV2-L uses ImageNet-22K pretrained weights fine-tuned on ImageNet-1K, while WideResNet-50-2 uses ImageNet-1K pretrained weights. DINOv3-L/16 uses the main-paper representations from `blocks.11` and `blocks.23`. For WideResNet-50-2, we use `layer2` and `layer3`, following the default PatchCore configuration in Anomalib. For SwinV2-L, we use `layers.1` and `layers.2`. The PatchCore coreset ratio, SAE width of 64 with TopK $k = 4$, annotation protocol, calibration split, intervention procedure, and corruption methodology are kept fixed.

Table 12: Vision backbone ablation over five categories. Deltas are AUROC percentage points relative to the corresponding original detector. Largest intervention gains are in bold.

Backbone	Source			Corrupted		
	Original	Suppression Δ	Joint Δ	Original	Suppression Δ	Joint Δ
DINOv3-L/16	0.8969	+1.36	**+2.86**	0.8428	+0.37	**+2.88**
SwinV2-L	0.8073	**+1.41**	+1.78	0.7177	**+1.02**	+1.22
WideResNet-50-2	0.9213	-0.19	-0.19	0.8297	-0.07	-0.07

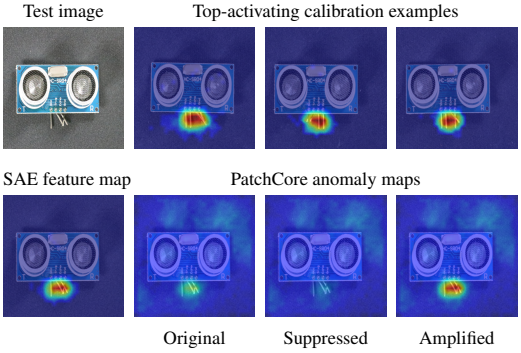
SwinV2 has a lower original AUROC than the other backbones, driven in part by Pushpins, where PatchCore performs at approximately chance level in both source and corrupted evaluation. On the same category, DINOv3-L/16 reaches 0.7666 and 0.6994 AUROC on source and corrupted data, respectively, while WideResNet-50-2 reaches 0.7286 and 0.6807.

The positive average effect obtained with DINOv3 also transfers to SwinV2 on these five categories. Joint intervention improves source and corrupted AUROC by 2.86 and 2.88 points with DINOv3, and by 1.78 and 1.22 points with SwinV2. In contrast, the average effect with WideResNet-50-2 is close to zero, with changes of -0.19 and -0.07 points.

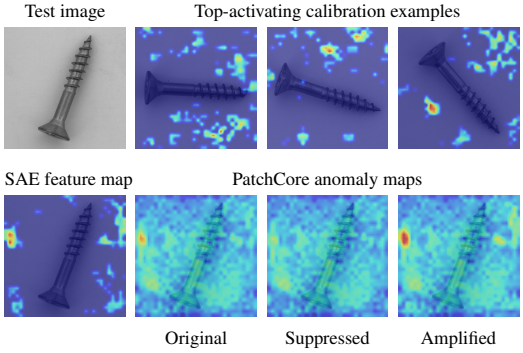
G ADDITIONAL QUALITATIVE EXAMPLES

Here we provide additional examples of SAE features and their intervention effects. Italic text gives the MLLM-generated feature description. For each feature, the top row shows a test image and three top-activating calibration examples. The bottom row shows the test SAE feature map and PatchCore anomaly maps before intervention, after suppression, and after 2× amplification. The three PatchCore anomaly maps for each example use a shared min-max color scale.

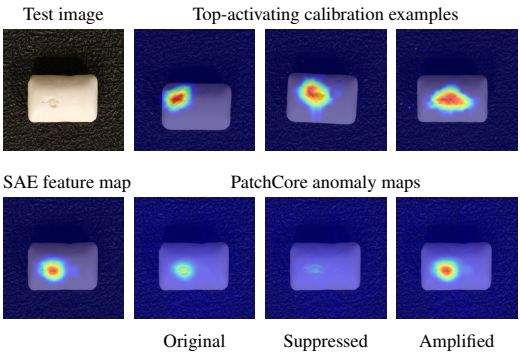
(a) Feature labeled as anomaly
VisA PCB1, SAE feature 3
"bent or angled connector pin"



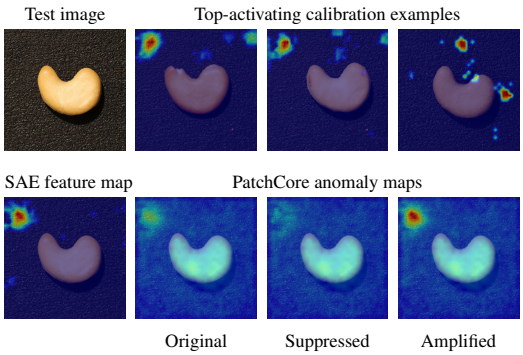
(b) Feature labeled as distractor
MVTec AD Screw, SAE feature 15
"faint isolated dark speck on the background"



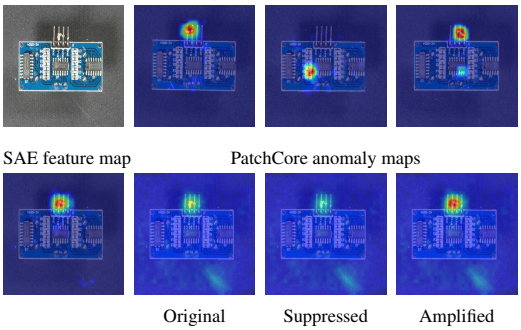
(c) Feature labeled as anomaly
VisA Chewing Gum, SAE feature 35
"torn, gouged chewing-gum surface with exposed brown interior"



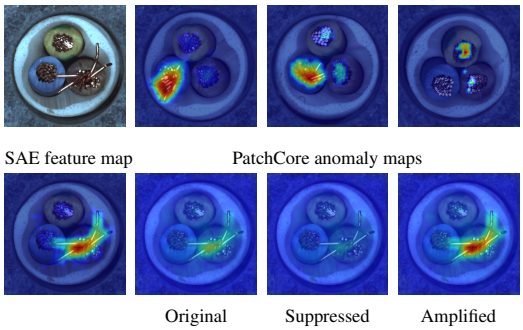
(d) Feature labeled as distractor
VisA Cashew, SAE feature 7
"glossy, strongly highlighted background fibers"



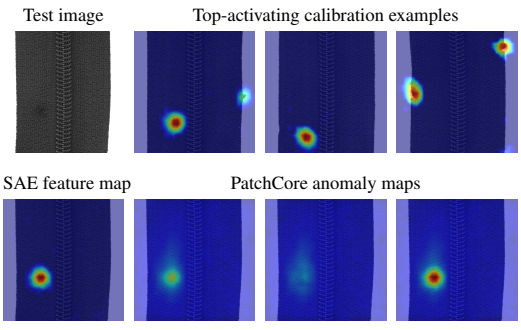
(e) Feature labeled as anomaly
VisA PCB2, SAE feature 44
"excess solder blobs or solder bridging on PCB components"



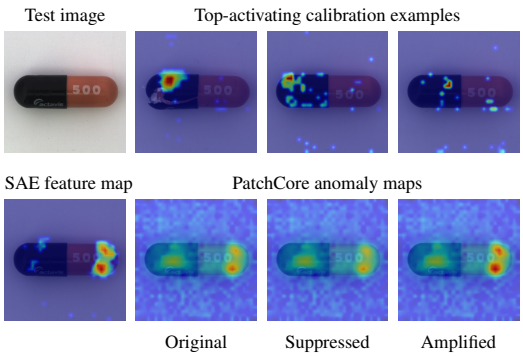
(f) Feature labeled as anomaly
MVTec AD Cable, SAE feature 8
"elongated protruding copper strands"



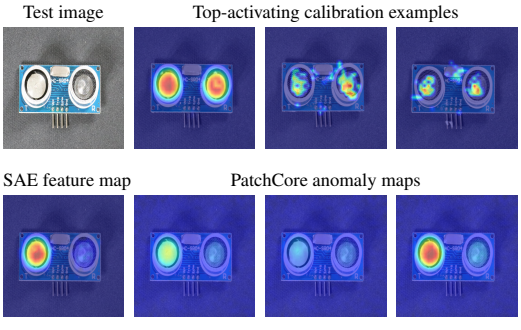
(g) Feature labeled as anomaly
MVTec AD Zipper, SAE feature 25
"tangled protruding threads or fibers"



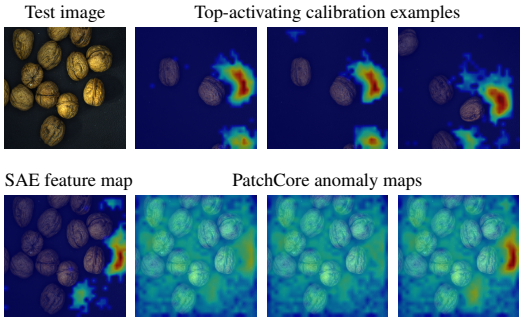
(h) Feature labeled as uncertain
MVTec AD Capsule, SAE feature 48
"capsule surface defects or markings"



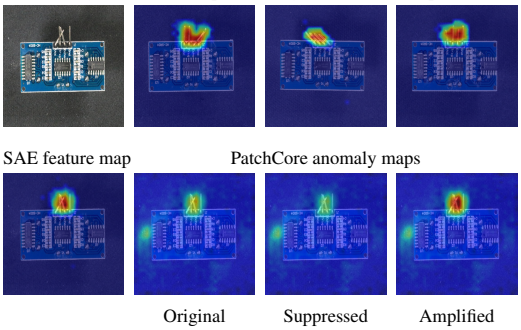
(i) Feature labeled as anomaly
VisA PCB1, SAE feature 16
"missing or severely damaged transducer mesh exposing a smooth inner disc"



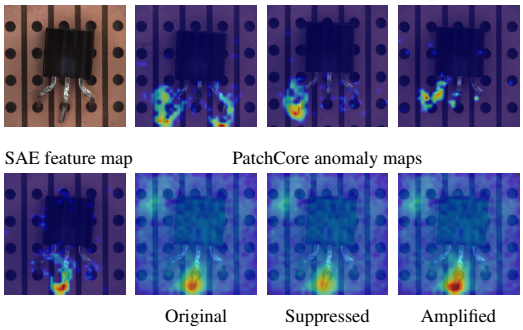
(j) Feature labeled as distractor
MVTec AD 2 Walnuts, SAE feature 26
"dark background with coarse diagonal ridges"



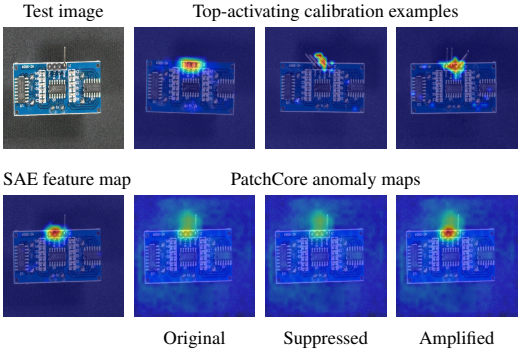
(k) Feature labeled as anomaly
VisA PCB2, SAE feature 50
"bent or misaligned header pins"



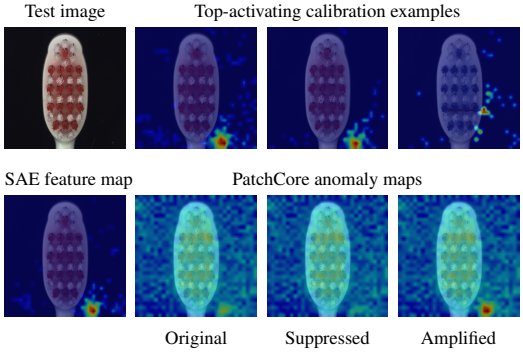
(l) Feature labeled as anomaly
MVTec AD Transistor, SAE feature 61
"transistor lead tip not inserted into a board hole"



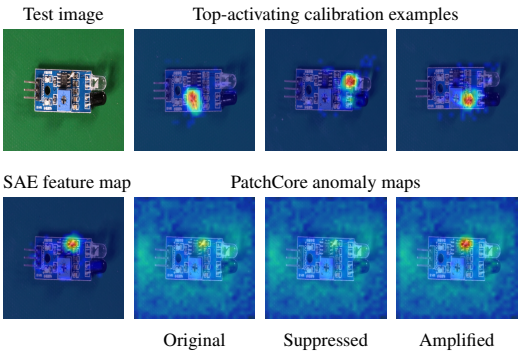
(m) Feature labeled as anomaly
VisA PCB2, SAE feature 9
"missing or severely displaced connector pins/exposed header pads"



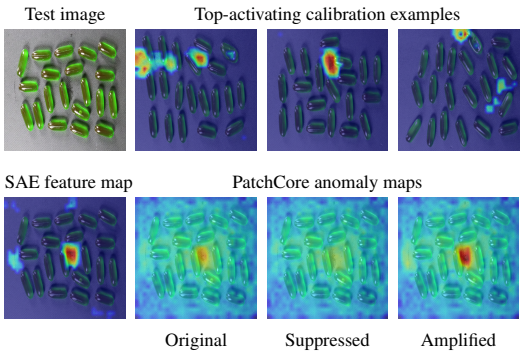
(n) Feature labeled as distractor
MVTec AD Toothbrush, SAE feature 58
"thin brown foreign filament or scratch on the dark background"



(o) Feature labeled as anomaly
VisA PCB3, SAE feature 44
"excess solder blobs or solder bridging on PCB components"

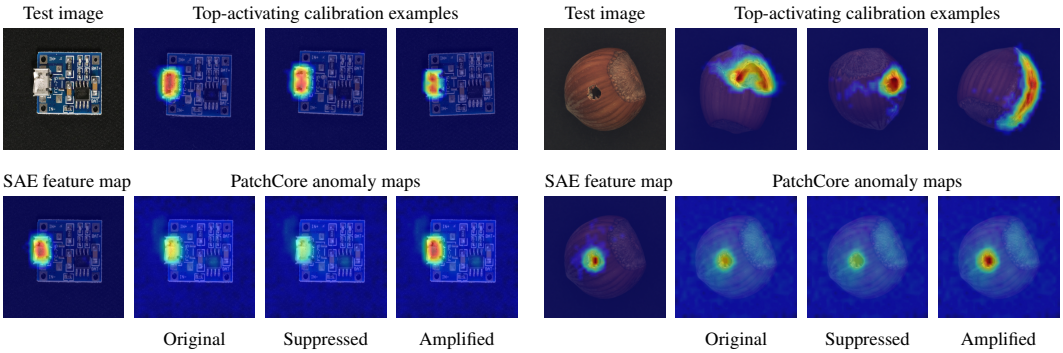


(p) Feature labeled as distractor
VisA Capsules, SAE feature 48
"dark smudge or stain on the background fabric"



(q) Feature labeled as anomaly
VisA PCB4, SAE feature 24
"missing or severely damaged USB connector shell"

(r) Feature labeled as anomaly
MVTec AD Hazelnut, SAE feature 10
"exposed pale inner shell along deep cracks or holes"



H EXAMPLE MLLM ANNOTATION PROMPT

System prompt

You are an expert in visual anomaly detection for industrial inspection. The dataset is RobustAD, category PCB. Official benchmark normality specification for this category: The PCB sub-dataset captures the challenges of finding subtle scratches, soldering melts, and missing parts which comprise of the most common defects encountered during inspection of Printed Circuit Boards in electronics and semiconductor manufacturing. Defects: Missing components, scratches, soldering melts

User prompt

We're studying a neuron in a sparse autoencoder used to interpret a PatchCore anomaly detector.

You will first be shown a reference grid of known-normal images.

You will then be shown high-activation context panels. Each panel contains the full image with the selected PatchCore region boxed, the neuron's spatial unit activation map, and a magnified crop with the same region boxed. All displayed activation maps use the same unit-specific color scale.

The reported activation is read out at the red-boxed PatchCore location; it does not describe the whole image. A zero-activation comparison means that the unit is inactive at the selected comparison location, even if another part of the image contains an anomaly or activates the unit. Compare the boxed regions, using the full images only as context.

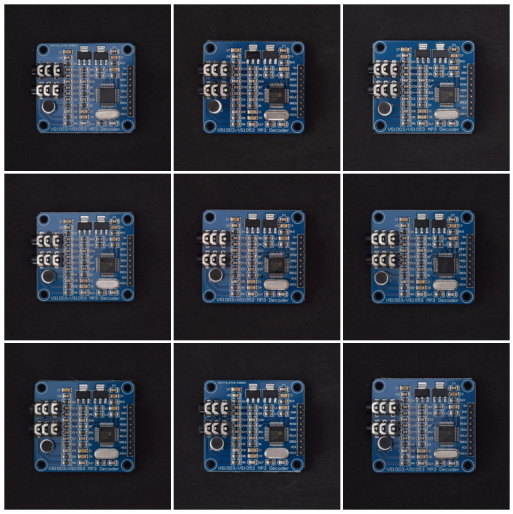
The colors in the unit activation map and the red boxes are visual aids; they are not present in the source image. Use the map only to locate activation, and infer visual content from the full image and high-activation region.

Treat the activation map and box as approximate spatial cues, since transformer tokens mix contextual information.

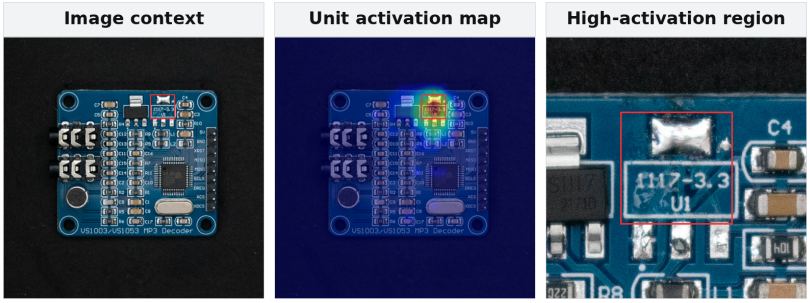
The final group contains zero-activation comparison panels. Each panel shows the complete source image with the selected PatchCore region boxed and the same region magnified alongside it.

Your task is to identify the common visual concept in the selected high-activation regions that is absent from the selected zero-activation comparison regions.

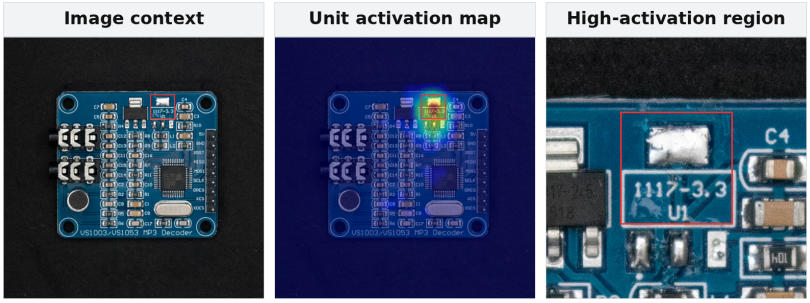
Known-normal reference grid



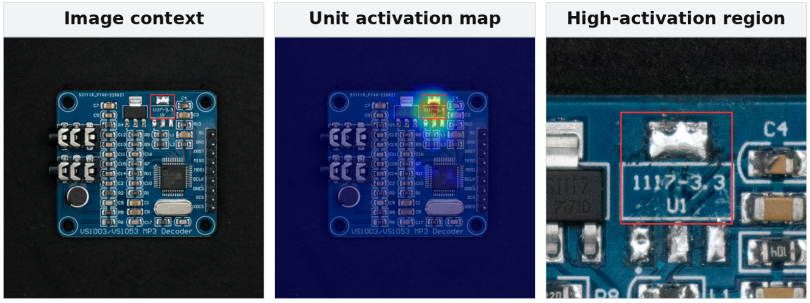
High-activation context panel 1, selected-location activation: 411.15



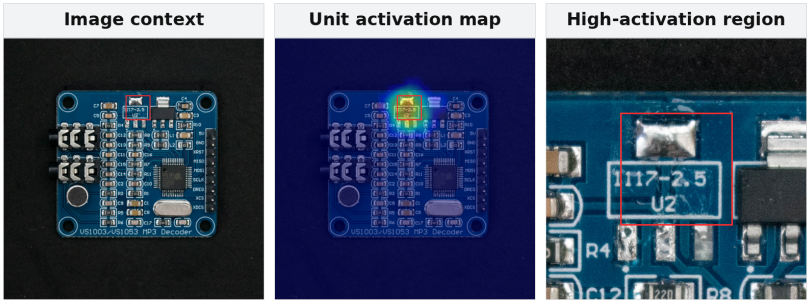
High-activation context panel 2, selected-location activation: 392.50



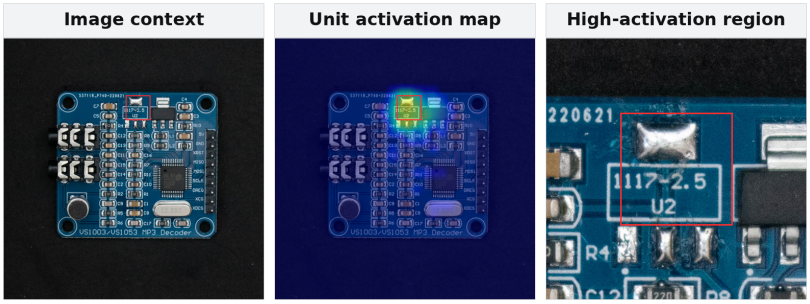
High-activation context panel 3, selected-location activation: 378.06



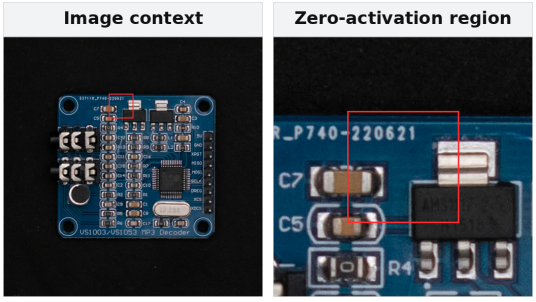
High-activation context panel 4, selected-location activation: 321.88



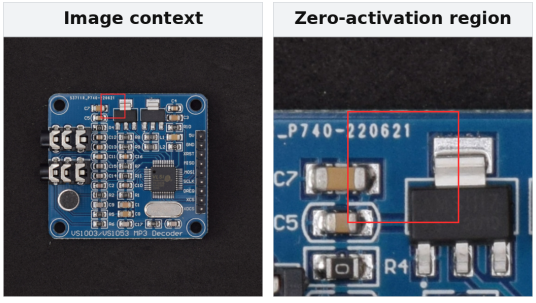
High-activation context panel 5, selected-location activation: 278.46



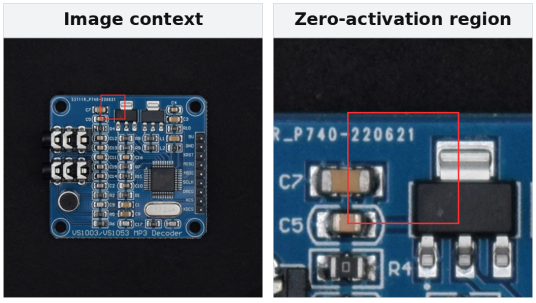
Zero-activation comparison panel 1



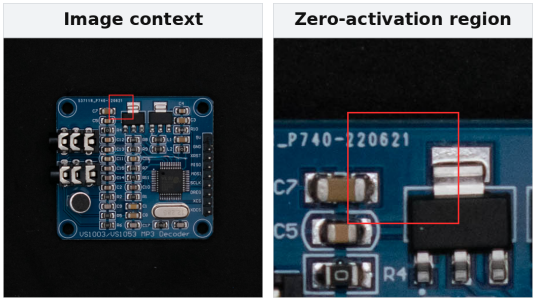
Zero-activation comparison panel 2



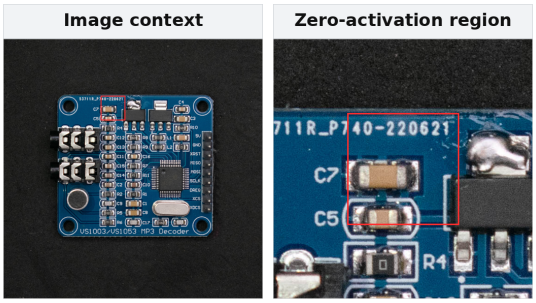
Zero-activation comparison panel 3



Zero-activation comparison panel 4



Zero-activation comparison panel 5



Now that you've analyzed the patches, provide just a concise concept name and its semantic type.

Name the visual attribute or combination of attributes that clearly appears to cause the activation, rather than describing the full patch or including weak incidental correlations. Avoid vague names like 'distinctive pattern'.

Classify the concept by its relevance to the inspection target, not by rarity alone. Use anomaly when it indicates an abnormality of the target or its expected configuration. Use distractor when it captures something unrelated to the target or an ordinary target variation. Use uncertain when the evidence is mixed or unclear.

Identify the localized visual feature that distinguishes the activating examples. Use the normal reference grid only to understand normal appearance, and do not classify from activation magnitude alone. Never name or classify colors, textures, or boundaries introduced by the unit activation map visualization.

Return a JSON object with exactly two keys: concept_name and semantic_type.