

Feature-Aware Test Generation for Deep Learning Models

XINGCHENG CHEN, Technical University of Munich, Germany and fortiss, Germany

OLIVER WEISSL, Technical University of Munich, Germany

ANDREA STOCO, Technical University of Munich, Germany and fortiss, Germany

As deep learning (DL) models are widely used in software systems, test generation plays a crucial role in assessing the quality of such models before deployment. To date, the most advanced test generators rely on generative AI to synthesize inputs; however, these approaches remain limited in providing semantic insight into the causes of misbehaviours and in offering fine-grained semantic controllability over the generated inputs. In this paper, we introduce DETECT, a feature-aware test generation framework for vision-based DL models that generates inputs by perturbing disentangled semantic attributes within the latent space. DETECT perturbs individual latent features in a controlled way and observes how these changes affect the model's output. Through this process, it identifies which features lead to behaviour shifts and uses a vision-language model for semantic attribution. DETECT enables semantically controlled perturbations in StyleGAN's disentangled latent space, allowing us to test whether a model generalizes under task-relevant feature changes and remains invariant under task-irrelevant ones. Empirical results across image classification and detection tasks show that DETECT generates high-quality test cases with fine-grained control, reveals distinct shortcut behaviours across model architectures (convolutional and transformer-based), and bugs that are not captured by accuracy metrics. Specifically, DETECT outperforms a state-of-the-art test generator in decision boundary discovery and provides more interpretable feature-level diagnoses than a leading spurious feature localization method in identifying robustness failures. Our findings show that fully fine-tuned convolutional models are prone to overfitting on localized cues, such as co-occurring visual traits, while weakly supervised transformers tend to rely on global features, such as environmental variances. These findings highlight the value of interpretable and feature-aware testing in improving DL model reliability, and refocus DL testing from class-level failure discovery to feature-level, semantically grounded diagnosis and repair.

CCS Concepts: • **Software and its engineering** → **Software testing and debugging**.

Additional Key Words and Phrases: Deep learning testing, test generation, explainable AI, spurious correlations, feature invariance

1 INTRODUCTION

Deep learning (DL) models such as convolutional neural networks (CNNs) and transformers have achieved impressive performance on many computer vision benchmarks [12], and they are widely deployed in many software systems nowadays. However, this success often fails to translate to real-world deployments, where models encounter inputs that differ, sometimes subtly, from those seen during training [26, 54, 80].

Authors' addresses: [Xingcheng Chen](#), xingcheng.chen@tum.de, Technical University of Munich, Munich, Germany, xchen@fortiss.org and fortiss, Munich, Germany; [Oliver Weißl](#), o.weissl@tum.de, Technical University of Munich, Munich, Germany; [Andrea Stocco](#), andrea.stocco@tum.de, Technical University of Munich, Munich, Germany, stocco@fortiss.org and fortiss, Munich, Germany.

These failures arise from two main challenges, namely *limited generalization* to unseen but valid inputs, and *lack of robustness* to spurious correlations that the model mistakenly relies on. Conceptually, these problems are related to the empirical risk minimization paradigm [69], which encourages DL models to optimize average performance over the training distribution, regardless of whether the features they exploit are semantically meaningful. As a result, models often learn *shortcut solutions*—spurious correlations tied to background, lighting, pose, or co-occurring objects—that work on the training set but fail when exposed to real-world data distributions [17, 22].

While deriving from the same root problem, in the literature, these two challenges have been investigated separately. While test generation techniques aim to evaluate model generalization by synthesizing new inputs that may uncover erroneous behaviours [53, 82], they often lack explainability and perturb multiple features in an entangled manner (e.g., by injecting global Gaussian noise or simultaneously modifying lighting, texture, and object semantics). Therefore, existing test generation techniques expose failures but entangle features and lack semantic attribution. Conversely, spurious feature analyses traditionally focused on robustness. They investigate whether models rely on invariant, task-irrelevant features, but typically through reactive post-hoc explanation tools such as saliency maps [7, 57]. This separation between input synthesis and feature diagnosis limits effective test generation and DL model improvements.

To fill this gap, in this paper, we propose DETECT, an explainable test generation framework that unifies *test generation* and *spurious feature identification* by testing behaviours of DL systems under *semantically controlled perturbations*. Our approach is based on the insight that both generalization and robustness concerns can be addressed through *feature-aware changes* in the disentangled latent space: by precisely manipulating individual semantic features, either relevant or irrelevant to the task, DETECT assesses not only if the DL model generalizes to novel inputs by varying the relevant features, but also whether it remains invariant under task-irrelevant changes.

To achieve this, DETECT leverages a style-based generative model trained on the same distribution as the system under test (SUT), which provides access to a disentangled latent space of semantic features. By perturbing latent dimensions, we generate realistic input variants with fine-grained changes in specific attributes. This enables *feature-aware test generation*, by perturbing task-relevant features to explore decision boundaries to uncover untested regions or modifying task-irrelevant features to assess prediction invariance and identify spurious dependencies. DETECT incorporates a *feature-aware test oracle*, which quantifies model response to controlled perturbations using a logit-based criterion. If the model’s prediction significantly shifts after modifying a semantically irrelevant feature, it is flagged as a robustness failure due to spurious correlation. Conversely, changes in response to relevant features are used to test generalizability.

In our empirical evaluation on image classification and detection tasks, DETECT outperforms existing test generation and spurious feature localization methods in revealing feature-sensitive failures, as well as robustness failures due to task-irrelevant features. DETECT reveals spurious correlations in high-accuracy models that standard

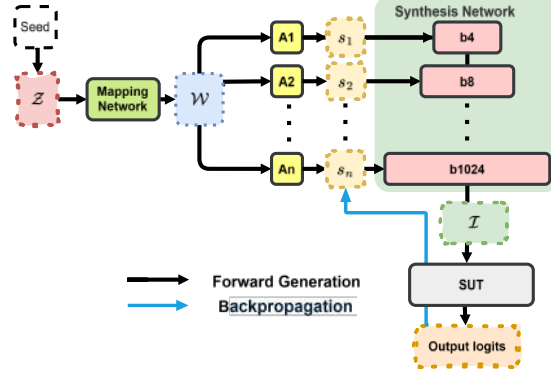


Fig. 1. **System Architecture of DETECT**: It comprises two main components, a StyleGAN generator and the SUT, and operates across three domains: the latent $\mathcal{S}$ -space, the image space $\mathcal{I}$, and the output space. It contains 1) a forward generation pass from an initial seed z mapped into a set of style vectors s , which synthesize an image input through synthesis blocks (b) and then pass to the SUT to obtain output logits; and 2) a backpropagation through SUT and synthesis network back to s .

accuracy metrics overlook, and distinguishes between different shortcut behaviours learned by weakly and fully supervised models. On fine-grained tasks, DETECT also serves as an effective general test generator, offering flexible applicability across model types and data domains.

Our contributions are as follows:

- 1) We propose DETECT, the first approach to unify latent-space test generation and spurious feature diagnosis under a feature-aware testing framework.
- 2) We introduce a feature-aware test oracle that dually checks *invariance* for irrelevant features (robustness) and explores *decision boundaries* for relevant features (generalization).
- 3) We show that generated tests are not only failure-inducing but also semantically attributable, and provide a proof-of-concept demonstrating how the generated feature-aware tests can be leveraged for model improvement (i.e., repair).

2 BACKGROUND

2.1 Generative AI in Deep Learning Testing

In the context of deep learning testing, generative AI techniques are used due to their ability to model complex, feature-rich data distributions and synthesize new functional inputs through latent space manipulation [45, 56]. Among these, Variational Autoencoders (VAEs) [14, 30], Generative Adversarial Networks (GANs) [44, 74], and diffusion models [49] are widely used for generative deep learning testing [44]. However, each comes with distinct limitations for feature-aware test generation

purposes. VAEs, while structurally interpretable, tend to produce low-fidelity and therefore low-quality input images [14, 30]. Conventional GANs suffer from entangled latent spaces, making fine-grained semantic control over specific generated features difficult [33]. Finally, while diffusion models can generate high-quality outputs, their iterative denoising process can make repeated test generation expensive [49]. A recent benchmark found that the average inference time of diffusion-based test generators was approximately 90 times that of GAN-based generators [44].

Among these, recent approaches based on style-based GANs (StyleGAN) [32–34] offer high controllability due to disentangled and interpretable latent spaces. These models enable precise control over high-level image characteristics, referred to as *features*, such as texture, pose, lighting, or background, through separate style and content layers [74]. This controllability makes StyleGAN particularly well-suited for feature-aware test generation, where the goal is to manipulate specific semantic attributes of an image. As such, the remainder of this section briefly details the StyleGAN architecture.

2.2 StyleGAN and Disentangled Latent Spaces

StyleGAN [32–34] differs fundamentally from other image generators and is specifically chosen as the backbone of DETECT due to its controllable and disentangled latent spaces. This architecture directly enables DETECT’s core capability: manipulating individual semantic features without affecting the rest of the image.

StyleGAN comprises two main components (see top left of Fig. 1): a mapping network and a synthesis network. The generation process begins with a randomly sampled latent code $z \in \mathcal{Z}$, which the mapping network then transforms into an intermediate latent space $\mathcal{W}$ that better captures disentangled image features. Subsequently, the synthesis network applies learned affine transformations to produce per-layer style vectors s , resulting in the StyleSpace ($\mathcal{S}$).

DETECT strategically leverages the $\mathcal{S}$ -space because these style vectors modulate the convolutional activations at each block, allowing the generator to control image features at different semantic and spatial levels. Early (coarse) layers influence global properties (e.g., pose, shape), while later (fine) layers affect localized details (e.g., texture, lighting). Each style vector in $\mathcal{S}$ consists of multiple *channels*, where each channel controls a distinct and semantically meaningful visual attribute, thereby enabling precise and localized image manipulations. While seminal computer vision works have successfully discovered interpretable, disentangled directions in other latent spaces (e.g., InterFaceGAN [58], LatentCLR [79], and Voynov et al. [70]), these semantics are encoded as multi-dimensional composite vectors. Precomputing and traversing these complex vectors is computationally suboptimal for systematic software testing. Conversely, specialized architectures that enforce mathematically orthogonal bases (e.g., EigenGAN [24]) provide decoupled control, but are typically limited to a small set of macro-level principal dimensions (e.g., global pose), lacking the granularity needed to expose subtle software defects. We utilize the $\mathcal{S}$ -space because its disentanglement is inherently channel-wise (axis-aligned). This structural property allows DETECT to

traverse and perturb distinct semantic features via single-channel modifications, seamlessly enabling our automated testing pipeline without the overhead of discovering composite directional vectors.

2.3 Motivating Examples

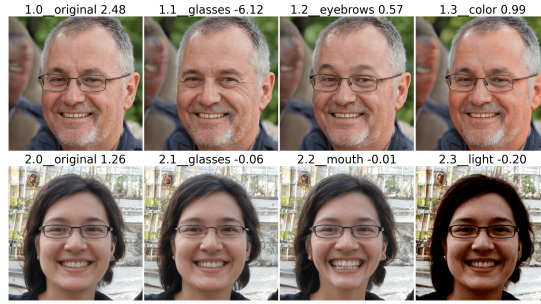


Fig. 2. Example perturbations in StyleSpace for glasses classification. Each row shows a base image (left) and variants generated by perturbing a single feature (e.g., glasses, eyebrows, or lighting). Titles: image index, perturbed feature, and model output logit.

Fig. 2 shows several examples related to the glasses detection task. By perturbing task-relevant features, one can control the presence or absence of glasses, resulting in a true negative prediction by the classifier on image 1.1 (where glasses are absent), but also a false negative prediction on image 2.1 (where instead glasses are present). Modifications on task-irrelevant features—such as eyebrows, facial expression, or sunlight—cause a drop in classifier confidence (such as images 1.2 and 1.3), and in some cases, even result in misclassifications (such as images 2.2 and 2.3). These findings highlight a critical challenge: not all influential features are task-relevant.

3 FEATURE-AWARE TEST ORACLE

A central component of our testing framework is a *feature-aware test oracle* that defines behavioural expectations based on the semantic relevance of input features. Unlike traditional test oracles that apply a uniform correctness criterion (e.g., *misclassification*) across all inputs, our oracle distinguishes between *task-relevant* and *irrelevant* features, assigning distinct testing objectives to each category. First, we define a feature as a semantically interpretable factor in the latent space, whose controlled manipulation leads to a measurable change in the generated input.

Formalization. Let $\mathcal{T}$ be a K -class classification task, and let $\mathbf{F}(x) \in \mathbb{R}^K$ be the model's output logits for input x , with $\mathbf{F}_t(x)$ representing the logit for the target class $t \in \{1, \dots, K\}$. Let $\phi \in \mathcal{F}$ be a controllable semantic feature of the input, manipulated through a generative model into a new input x' .

Definition 1: Task-Relevant Feature

A feature ϕ is **task-relevant** with respect to task $\mathcal{T}$ (denoted $\phi \in \mathcal{F}_{\text{rel}}^{\mathcal{T}}$) if perturbing ϕ induces a meaningful change in the ground truth label or alters the semantics necessary for correct prediction under $\mathcal{T}$. Let $x \sim_{\phi} x'$ denote that x and x' differ only in feature ϕ . Then, a feature ϕ is task-relevant if:

$$\exists (x, x') \text{ such that } x \sim_{\phi} x', \quad \text{label}(x) \neq \text{label}(x'). \quad (1)$$

Definition 2: Task-Irrelevant Feature

A feature ϕ is **task-irrelevant** (denoted $\phi \in \mathcal{F}_{\text{irr}}^{\mathcal{T}}$) if changes in ϕ do not alter the ground-truth label under task $\mathcal{T}$, and are semantically non-essential to the prediction objective. Formally,

$$\forall (x, x') \text{ such that } x \sim_{\phi} x', \quad \text{label}(x) = \text{label}(x'). \quad (2)$$

Definition 3: Spurious Feature

A feature $\phi \in \mathcal{F}_{\text{irr}}^{\mathcal{T}}$ is **spurious** if there exists a pair of inputs that differ only in ϕ , and this change exercises a significant influence on the model's prediction. Formally, ϕ is spurious if:

$$\exists (x, x') \text{ such that } x \sim_{\phi} x', \quad \Delta_{\text{logit}}(\phi) = |\mathbf{F}_t(x') - \mathbf{F}_t(x)| > \tau, \quad (3)$$

where τ is a predefined confidence threshold.

Our oracle supports two complementary testing goals:

Evaluating Generalization via Oracle misclassification. It applies to relevant features for behaviour exploration (i.e., DNN generalization). For $\phi \in \mathcal{F}_{\text{rel}}^{\mathcal{T}}$, we perturb inputs to explore semantic behaviours. Prediction changes here are expected and indicative of the model's sensitivity to meaningful feature variation. Misclassifications near these transitions signal overfitting or under-generalization.

Evaluating Robustness via Oracle confidence. It applies to irrelevant features for DNN robustness assessment. For $\phi \in \mathcal{F}_{\text{irr}}^{\mathcal{T}}$, the model should behave invariantly. Thus, this oracle monitors $\Delta_{\text{logit}}(\phi)$ as a proxy for overreliance on non-causal features. Large shifts indicate spurious behaviour.

In practice, due to the hierarchical and interdependent nature of some features, the latent space of the generator cannot be perfectly disentangled. We select $\mathcal{S}$ space as the manipulation space, as it achieves a disentanglement score of 0.75, much higher than 0.54 measured from $\mathcal{W}$ space [77] (where 1.0 represents perfect separation of semantic attributes). In this sense, $\mathcal{S}$ space represents the most disentangled and controllable space currently attainable for our purpose. To ensure methodological rigor, we adopt a conservative attribution rule: when a feature modification introduces a clear relevant

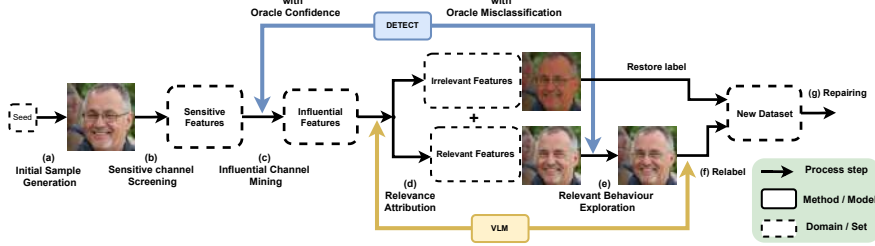


Fig. 3. Sequential Pipeline of DETECT. Sensitive channels are mined with a confidence-drop oracle, attributed as relevant or irrelevant by the VLM, explored near the decision boundary when relevant, and relabeled to form a repair dataset.

semantic change but also causes small incidental adjustments, we treat the feature as relevant and ignore such minor co-effects. For example, modifying the presence of glasses may slightly affect nearby shading or eyebrow shape.

4 METHODOLOGY

Fig. 3 illustrates the workflow of our proposed framework, which we call DETECT. DETECT first samples a random latent seed, which is passed through a StyleGAN trained on the same domain as the SUT to generate an image (step Initial Sample Generation). To understand which channels in each latent feature s_i affect the model's prediction, DETECT incorporates XAI methods to estimate the sensitivity of each channel on the target logit (step Sensitive Channel Screening). Specifically, we employ gradient-based approaches such as Gradient Saliency [60] and SmoothGrad [61], as well as the black-box Finite-Difference Approximation [48]. Then, DETECT invokes an oracle-aware perturbation loop twice with different oracles. In the first iteration (step Influential Channel Mining), DETECT perturbs all sensitive features along the direction that reduces the SUT's confidence and collects the *influential* features that cause significant changes in the model's output. DETECT then identifies whether these are task-relevant using a Vision Language Model (VLM) (step Relevance Attribution). If a feature is found to be semantically irrelevant yet influences the output, DETECT labels it as *spurious*. On the other hand, for features deemed relevant, DETECT performs a second perturbation guided by the Oracle Misclassification to explore the SUT behaviours (step Relevant Behaviour Exploration). The second perturbation is optimized via hill climbing until decision-boundary tests are obtained. These new inputs are then relabeled by the VLM, producing a test suite that captures both relevant and spurious feature variations.

In the remainder of the section, we describe each step in greater detail.

4.1 Initial Sample Generation

First, a random latent seed z is generated and passed through a StyleGAN mapping network to obtain an intermediate latent vector $w \in \mathcal{W}$ (Fig. 1). This vector is then

transformed into a set of style vectors s_i , each of which modulates a specific synthesis block and spatial resolution during image generation. Then, the generated image $\mathcal{I}$ is fed into the SUT to obtain class logits. DETECT uses the logit output of the target class because it provides a direct, unnormalized measure of the SUT's confidence, allowing precise assessment of the model's sensitivity to controlled changes in specific latent features.

4.2 Sensitivity Screening Methods

Unlike most search-based latent space test generation methods exploring the entire high-dimensional feature space [44, 74] (typically spanning thousands of latent dimensions), DETECT employs XAI-based sensitivity estimation to efficiently screen sensitive features and directions, reducing the subsequent perturbation search space.

We define the sensitivity of the SUT's target logit $y[t]$ to the i -th style vector s_i , where $G(s)$ denotes the synthesis network that synthesizes the image from the style vector $s = \{s_1, \dots, s_n\}$ and $s_i \in \mathbb{R}^d$ (the length of style vector d varies among different layers), i.e.,

$$\alpha_i^{(t)} = S(y[t], s_i) = S((F_t \circ G)(s), s_i) \in \mathbb{R}^d, \quad (4)$$

where $y[t]$ denotes the logit of SUT for the target class t and the sensitivity α_i has the same dimension as s_i .

In this work, we adopt three established XAI methods [11], applied not to the image space, but to the disentangled latent space.

Gradient Saliency [60]. It computes the sensitivity of the target logit $y[t]$ with respect to each style vector s_i by backpropagation through the composed function $F_t \circ G$, i.e.,

$$\alpha_i^{(t)} = \frac{\partial y[t]}{\partial s_i} = \frac{\partial (F_t \circ G)(s)}{\partial s_i}. \quad (5)$$

This serves as a first-order approximation of the local sensitivity of the output to each style component.

SmoothGrad [61]. While pure gradients are often noisy and sensitive to local variations in the input space, we adopt SmoothGrad to improve stability by averaging gradients over noise-perturbed inputs. We sample N noisy style vectors $\{s^{(j)} = s + \epsilon^{(j)}\}_{j=1}^N$ with $\epsilon^{(j)} \sim \mathcal{N}(0, \sigma^2 I_d)$, and then estimate the sensitivity of style vector s_i to output logit $y[t]$ as:

$$\alpha_i^{(t)} = \frac{1}{N} \sum_{j=1}^N \frac{\partial y[t]}{\partial s_i^{(j)}} = \frac{1}{N} \sum_{j=1}^N \frac{\partial (F_t \circ G)(s^{(j)})}{\partial s_i^{(j)}}. \quad (6)$$

Finite-Difference Approximation [48]. This quantifies the local effect of each latent channel via output variation by computing a gradient-free sensitivity approximation based on finite differences. This method is applicable even when the composed model $F \circ G$ is non-differentiable or analytically inaccessible. Let $\hat{s}^{(i,c)} = (s_1, \dots, s_i + \Delta e_c, \dots, s_n)$, where e_c is the c -th standard basis vector in $\mathbb{R}^{d_i}$. The sensitivity

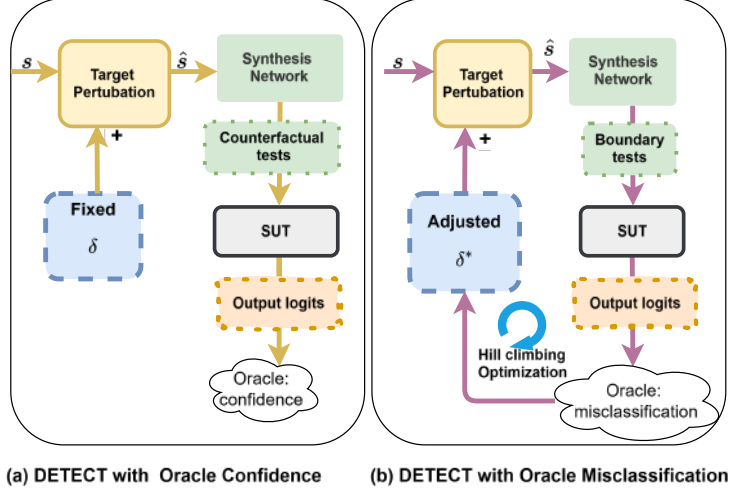


Fig. 4. DETECT with Oracle Confidence and Misclassification.

is estimated as:

$$\alpha_{i,c}^{(t)} = \frac{(\mathbf{F}_t \circ \mathbf{G})(\hat{s}^{(i,c)}) - (\mathbf{F}_t \circ \mathbf{G})(s)}{\Delta}. \quad (7)$$

After computing the sensitivity scores, DETECT retrieves a channel-wise attribution map for each layer in the synthesis network, $\alpha_i^{(t)} \in \mathbb{R}^{d_i}$ for each layer i . Since each style vector can have up to a thousand latent channels, only the top- k channels with the highest absolute sensitivity scores are retained for further analysis. The ranking is performed strictly at the channel level within each individual style vector s_i , which is formally given by:

$$C_i = \left\{ c \in \{1, \dots, d_i\} \mid \text{rank} \left(\left| \alpha_{i,c}^{(t)} \right| \right) \leq k \right\},$$

where the $\text{rank}(\cdot)$ function sorts the absolute sensitivity scores $\alpha_{i,c}^{(t)}$ of the c -th channel in layer i in descending order. The overall candidate set of sensitive channels, denoted as $C_{\text{sensitivity}}$, is then constructed as the union across all layers: $C_{\text{sensitivity}} = \bigcup_i \{(i, c) \mid c \in C_i\}$. Channels outside this set are considered inactive or behaviourally negligible with respect to the SUT. While we use absolute sensitivity for ranking, we preserve the sign of each score. In fact, the directionality of the attribution (i.e., whether increasing or decreasing a channel pushes the output logit up or down) is used by the perturbation procedure to apply signed directional changes to the most influential channels.

4.3 Perturbation Loop

Once sensitive channels have been identified, DETECT invokes oracle-specified perturbations to evaluate their behavioural influence, as outlined in stages (c) Influential Channel Mining and (e) Relevant Behaviour Exploration in Fig. 3. This two-stage

Algorithm 1 CHANNELPERTURB(s, F, C, O, τ)

Require: Style vector s , classifier F , channel set C , oracle O , threshold τ

```

1: Initialize perturbed result set  $\mathcal{R} \leftarrow \emptyset$ 
2: for each  $(l, c)$  in  $C$  do
3:   Perturb  $\hat{s}_l[c] = s_l[c] + \delta$ 
4:   Generate  $\hat{x} = G(\hat{s})$ ,  $\hat{y} = F(\hat{x})$ 
5:   if  $O = \text{confidence}$  and  $\Delta_{\text{confidence}} > \tau$  then
6:     Add  $(l, c, \hat{x})$  to  $\mathcal{R}$ 
7:   else if  $O = \text{misclassification}$  and  $\mathcal{L}(\hat{y}) \neq \mathcal{L}(y)$  then
8:     Refine  $\delta^*$  via hill climbing to decision boundary
9:     Add  $(l, c, \hat{x}^*)$  to  $\mathcal{R}$ 
10:  end if
11: end for
12: return  $\mathcal{R}$ 

```

perturbation is guided by the feature-aware oracles described in Section 3. Fig. 4 provides an overview of these two configurations. In the first configuration (a), the perturbation δ is fixed, and the Oracle Confidence guides the identification of influential features by measuring the change in output logits. In the second configuration (b), the perturbation δ is iteratively adjusted through hill-climbing optimization, using misclassification as the oracle, to generate tests located near the decision boundary. The detailed algorithmic steps of this loop are provided in Algorithm 1.

Algorithm 1 illustrates the test generation by perturbing latent channels under a specified oracle. Here, C denotes a set of selected channels (identified via sensitivity analysis or semantic relevance), and O specifies the behavioural oracle, either confidence or misclassification.

The main loop of the algorithm enumerates each channel $(l, c) \in C$, and then applies a perturbation δ on each selected channel $s_{l,c}$ scaled by a fixed step size ϵ (Line 3). The step size ϵ is chosen to induce a visible semantic change in the generated image, while avoiding artifacts or unrealistic distortions [44, 77]. The step size ϵ is determined by the scale of the latent space, independent from the dataset or the downstream SUT, since the values in $\mathcal{S}$ space after affine transforms lie in stable numeric ranges [33]. The perturbation step δ is constructed to reduce the model's confidence in the current prediction. Its formulation differs slightly between binary and multiclass classification, as shown below:

$$\delta = \begin{cases} -\epsilon \cdot \text{sign}(\alpha_{i,c}^{(t)}) \cdot \text{sign}(y[t]), & \text{binary,} \\ -\epsilon \cdot \text{sign}(\alpha_{i,c}^{(t)}), & \text{multiclass.} \end{cases} \quad (8)$$

This design ensures that the perturbation moves in the direction opposite to the gradient-derived influence, optionally adjusting for logit polarity in binary classification.



Fig. 5. Example of Image Input to VLM: (left) the original generated image; (middle) the perturbed image generated by altering a single latent feature; and (right) a difference mask highlighting the localized regions where the semantic change occurred.

The perturbed latent $\hat{s}$ is then used to synthesize the perturbed image $\hat{x} = G(\hat{s})$, and obtain the system's response $\hat{y} = F(\hat{x})$ (Line 4). The logit $\hat{y}$ is analyzed according to the selected oracle. Under the confidence-based oracle, DETECT records the perturbed image if the drop in the target logit exceeds a predefined threshold, indicating that the SUT's prediction relies on the perturbed feature (Lines 5-6). Under the misclassification-based oracle, DETECT checks whether the predicted label changes (Lines 7-9). We assume $\mathcal{L}(\cdot)$ returns the predicted class. For binary classification, $\mathcal{L}(\cdot)$ checks the sign in the single logit; for multiclass, it takes the top-1 class. If misclassification occurs, DETECT refines the perturbation δ using a hill-climbing strategy to find the minimal modification required to cross the decision boundary, thereby producing more precise boundary examples $\hat{x}^*$.

Algorithm 1 is executed twice: first, to probe all sensitive channels using the confidence-drop oracle; and second, only on task-relevant candidates to explore the model's behaviour boundaries using the misclassification oracle. Specifically, DETECT first obtains channel-wise sensitivities using one of the methods in Section 4.2 and then extracts the top- k most sensitive channels c by sorting their absolute sensitivity scores to form a candidate set $C_{sensitivity}$. Then we apply the first-stage CHANNELPERTURB with the confidence-based oracle to isolate latent channels whose perturbation causes significant behavioural shifts. These are referred to as *influential* channels. However, at this point, DETECT does not yet distinguish whether the altered features are task-relevant or irrelevant, which is done in the subsequent relevance-attribution step.

4.4 Relevance Attribution, Relabel, and Repair

In this section, DETECT incorporates a vision-language model (VLM) as an automated semantic-annotation proxy to establish the ground truth. Specifically, it functions to identify semantic feature changes and provide automated annotations for the generated test cases, as illustrated in steps (d) Relevance Attribution and (f) Relabel in Fig. 3. The VLM provides text-aligned supervision for generated images, enabling DETECT to automatically determine whether each perturbation expresses a task-relevant concept or a spurious one.

DETECT's relevance attribution module systematically categorizes each feature ϕ into $\mathcal{F}_{rel}^T$ or $\mathcal{F}_{irr}^T$ in a data-driven manner. Specifically, as shown in Fig. 5, the image input

to VLM is composed of three concatenated images: (1) the original image, (2) the image after a single latent-direction change, and (3) a difference mask that highlights where the change occurs. The prompt follows a zero-shot, role-playing paradigm, instructing the VLM only to judge changes to the relevant attribute, and constraining output to strict JSON-formatted responses (see [Appendix B](#) for the full prompt examples). For each channel in the influential feature set, DETECT aggregates VLM decisions across multiple counterfactual samples and assigns a label by majority voting. Channels whose perturbations align with the task semantics are labeled as relevant, while those inducing visually noticeable but task-irrelevant changes are labeled as spurious. For instance, in [Fig. 5](#), a perturbation primarily alters the eyebrow region without affecting the presence of glasses; such a change is identified as task-irrelevant.

Beyond this semantic supervision, DETECT also exploits the inherent spatial hierarchy of StyleGAN. Empirically, channels from coarse-to-middle layers—responsible for high-level attributes such as shape, pose, or object identity—tend to correspond to relevant semantics, whereas fine-layer perturbations often lead to superficial changes like lighting or color shifts [33].

For the subset of relevant features, the VLM is employed again to relabel the decision-boundary images produced in the oracle misclassification loop, since the original labels are no longer preserved (see [Appendix B](#) for the full prompt examples). For spurious features, DETECT restores the original labels, consistent with our assumption of feature invariance.

Finally, DETECT incorporates both relevant and spurious features into a repaired dataset for fine-tuning of the SUT to improve both generalizability and robustness. The fine-tuning follows a lightweight strategy using a small learning rate and limited epochs, allowing the model to adapt to corrected feature-label associations without overfitting to the generated data.

With DETECT’s methodology established, the next step is to evaluate its effectiveness and practical utility. To provide a clear context for our experimental design, our evaluation aims to address four core aspects: (1) finding the best-performing pipeline configuration, (2) assessing the capability to explore generalized model behaviours, (3) detecting SUT failures driven by spurious correlations, and (4) demonstrating the potential to repair the model using the generated tests.

5 EMPIRICAL STUDY

5.1 Research Questions

RQ₁ (configuration): What configuration of DETECT yields the most effective and efficient feature-aware testing?

RQ_{1.1}: Which sensitivity screening strategy best identifies impactful latent channels?

RQ_{1.2}: What is the optimal number of channels to balance coverage and efficiency?

RQ_{1.3}: How do different VLMs affect the accuracy and cost-efficiency of feature attribution and labeling?

RQ₂ (generalization via behaviour exploration): How effective is DETECT at exploring decision boundaries on relevant features to reveal generalization failures?

RQ₃ (robustness via spurious feature detection): How effective is DETECT at evaluating model invariance and robustness against task-irrelevant features?

RQ₄ (model repair): Do the generated test cases contribute to improving the SUT?

RQ₁ investigates how to best configure DETECT for effective and efficient feature-aware testing. We examine which sensitivity screening strategy most effectively identifies impactful latent channels for targeted perturbation, explore how many channels should be perturbed to balance coverage and efficiency, and analyze how different VLMs influence the accuracy and cost-efficiency of feature attribution and labeling. RQ₂ evaluates DETECT's effectiveness in uncovering behavioural failures in the system under test (SUT). RQ₃ focuses on assessing whether DETECT can reveal failures caused by spurious or non-robust features. Finally, RQ₄ investigates whether the generated test cases can be leveraged to improve the SUT through retraining.

5.2 Metrics used for Analysis

For RQ_{1.1}, we compare different configurations of DETECT in terms of their runtime (in seconds) and the extent to which they alter the SUT's behaviour.

For RQ_{1.2}, we measure the number of features in each layer that can affect the behaviour of the SUT.

For RQ_{1.3}, we measure the agreement rate between the VLM and multiple human annotators. Besides, to mitigate inconsistency among different annotators, we evaluate consensus agreement against a majority-vote human label, where aggregated labels from multiple annotators represent the consensus reference label. In addition to correctness, we also assess the monetary cost per evaluation and average response time to measure their computational and economic efficiency.

For RQ₂, we evaluate the runtime of the compared methods in seconds. To assess whether structural information is preserved in the generation, we compute the MS-SSIM [73] score, which extends SSIM by computing similarity across multiple scales via iterative downsampling. This multi-scale structure is better aligned with human perception than regular SSIM. Since MS-SSIM is less sensitive to color variation, we complement it with the normalized $\tilde{d}_2$ -Image distance defined as $\tilde{d}_2(x_i, x_j) = L_2(x_i - x_j) / L_2(\mathbf{1}^x)$, where $\mathbf{1}^x$ is an all-ones matrix with the same shape as x . This metric aims to capture the overall magnitude of change, including color shifts. Finally, we evaluate the proximity to the decision boundary to assess the effectiveness of the generated test cases in challenging the model's predictions.

For RQ₃, we propose a simple metric to quantify how much a model relies on task-relevant features, i.e., the proportion of influential channels that are task-relevant:

$$R_{\text{Relevance}} = \frac{|\text{Task-Relevant Influential Features}|}{|\text{All Influential Features}|}$$

This metric captures the alignment between a model's decision-making process and the task-defined semantics. A higher value indicates that the model's behaviour is

driven more by meaningful features, whereas a lower value suggests susceptibility to spurious correlations.

5.3 Objects of Study

5.3.1 Datasets and Models. To ensure the generality and robustness of our evaluation, we selected three diverse feature-rich datasets, each paired with a corresponding SUT. **Face Attributes (CelebA [42]).** This dataset is labeled with 40 facial attributes, with each attribute as an independent binary classification task, although our subsequent analyses primarily focus on the eyeglasses and gender attributes. We evaluate two different SUTs: (1) *ResNet50* [23], consisting of a pretrained backbone followed by a custom fully connected classification layer. The entire model is fully fine-tuned on CelebA to predict all 40 attributes. It achieves 99.70% test accuracy on the eyeglasses attribute and 98.17% on gender. (2) *SWAG ViT* [46], a Vision Transformer with a frozen pretrained backbone and a custom two-layer classification head, trained in a weakly supervised manner on CelebA attributes. This model is chosen for its reported robustness against spurious features. It achieves a test accuracy of 99.17% on the eyeglasses attribute and 98.91% on the gender attribute.

Dog Breed Classification (Dogs [40]). We use a publicly available dataset from Kaggle [40], which includes labeled images of 78 dog breeds. This dataset is used to evaluate multi-class classification performance. Only dog images were retained for our multi-class classification task. The model under test is a pretrained ReXNet-150 [21], which achieved a test accuracy of 87.9%.

Car Detection (COCO [41]). We evaluate object detection performance on the car class. The supervised model under test is YOLOv8, implemented via the official Ultralytics framework [29]. The model is pretrained on the COCO dataset, and we directly apply this pretrained model to detect cars in our images without any fine-tuning. On the LSUN Cars dataset [78], we report an accuracy of 97.63% when including trucks, since some vehicles are ambiguous and may be classified as either car or truck; accuracy considering only cars is 78.83%.

The rationale behind this selection is twofold: (1) Task Diversity: The selected datasets encompass a broad spectrum of computer vision tasks, including high-fidelity binary attribute classification (CelebA), fine-grained multi-class classification (Dogs), and object detection (COCO). (2) Architectural Variety: To demonstrate that DETECT generalizes across varying model architectures, sizes, and training paradigms, we selected a diverse and representative set of SUTs. ResNet-50 (25M parameters) represents the classic fully fine-tuned CNN. In contrast, SWAG ViT (632M parameters) introduces a modern, large-scale Vision Transformer trained in a weakly-supervised manner. Furthermore, ReXNet-150 represents an efficient, lightweight architecture, and YOLOv8 represents state-of-the-art dense prediction models. Together, these encompass a wide spectrum of model capacities, architectural designs, and tuning strategies deployed in practice.

5.3.2 Generator. Each dataset is paired with a pretrained StyleGAN generator. We use the official StyleGAN2 models pretrained on FFHQ [34], LSUN Dog [78], and

LSUN Car [78], and further fine-tune each generator on the corresponding downstream dataset using a few thousand task-specific images and adaptive discriminator augmentation (ADA). This setup follows common practice in latent-space test generation methods [44, 49], where pretrained generators are preferred because their latent spaces capture a wider range of features learned from large and diverse datasets. As demonstrated in foundational studies on generative transfer learning [31, 71], high-fidelity GANs are data-hungry. Directly training a generator from scratch on the SUT’s dataset, which is often constrained in size and diversity, typically leads to mode collapse and an impoverished feature space. In contrast, fine-tuning a model pretrained on a large-scale dataset effectively transfers a rich, disentangled visual prior to the target domain [71]. To preserve this structural prior while adapting to the target dataset, we implement a strict transfer learning configuration: we freeze the early layers of the discriminator (e.g., the first 10 layers) and apply a lower learning rate (1×10^{-4}). This setup ensures the generator smoothly adapts to the domain-specific semantics (e.g., localized styles and textures) without distorting its core generative manifold.

5.3.3 Configurations. Regarding configuration of sensitive channel screening (RQ_{1.1}), we evaluate three methods in DETECT, including two white-box approaches, *Gradient Saliency (Grad)* and *SmoothGrad*, and one black-box approach, the *Finite-Difference Approximation (FDA)*. We chose these methods for different application requirements and computational complexity. *Grad* requires a single backward pass through the SUT and generator. *SmoothGrad* increases this cost by performing N backward passes, each on a perturbed version of the input, to generate more robust attributions through averaging. Following prior work [61], we set $N = 10$. In contrast, *FDA* is gradient-free and suitable for black-box settings. Instead, it perturbs each individual channel of style vectors and observes the resulting changes. Given that our StyleGAN model contains 9,088 channels, *FDA* performs 9,088 SUT forward passes to approximate relevance. Based on the empirical results of RQ_{1.1}, the best-performing XAI method is selected for the subsequent studies. Consequently, RQ_{1.2} inherits this XAI configuration to conduct its in-depth layer-wise analysis without introducing new variables. Besides, to support relevance attribution and relabel procedures (RQ_{1.3}), we evaluate 7 VLMs covering different model families, sizes, and reasoning capabilities. The selected models include Qwen-VL [66] (8B-Thinking, 30B-A3B-Thinking, and max), Mistral VLM variants [28] (small-3.1-24b-instruct, medium-3.1), and OpenAI models [2] (gpt-o4-mini, gpt-4.1).

Regarding RQ₂, to ensure a fair comparison with conventional test generation methods, which merely aim to find misbehaviour and lack the mechanism to distinguish between relevant and irrelevant features, it is necessary to include a simplified version of our framework, denoted as DETECT*. Unlike the full pipeline, DETECT* omits the relevance attribution step and perturbs all channels identified as sensitive in the initial screening. It relies solely on the misclassification-based oracle and thus functions as a general-purpose test generator, aimed at exposing misbehaviours under small, single-feature perturbations. The full DETECT pipeline (perturbing only relevance-filtered

channels under the misclassification oracle) is reported for the CelebA tasks, where relevance filtering leaves enough task-relevant channels for a meaningful comparison.

Regarding RQ₃, we utilize the full DETECT pipeline for all datasets to detect spurious features. While general-purpose VLMs may lack the expert-level taxonomy required for fine-grained breed identification, we found them reliable for distinguishing between breed traits and spurious environmental factors (e.g., lighting, background).

We constrain the automated repair task (RQ₄) to the CelebA dataset, as a proof-of-concept demonstrating the actionable potential of the generated tests for model improvement. For YOLO, besides the class label, the additional bounding box labels of generated images are required, which are currently outside the scope of this task. For the Dog dataset, while the VLM might be capable of labeling, the fine-grained breed distinctions make it practically difficult for non-expert annotators to validate rigorously.

5.3.4 Comparison Baselines. Although no existing method jointly addresses the dual objectives of our approach, we compare DETECT to representative baselines that align with each individual testing objective, latent-space test input generation, and spurious feature identification.

Mimicry. A state-of-the-art latent-space test generation method, that exploits the $\mathcal{W}$ -space of StyleGAN. It interpolates between w -latent vectors from different class instances and uses population-based optimization to tune the interpolation weights. The resulting synthesized images form test cases positioned near the SUT's decision boundary [74]. Like many other latent-space test generators, the entire latent vector is manipulated during this process, confounding relevant and irrelevant features. In contrast, DETECT operates in the more disentangled $\mathcal{S}$ -space and efficiently screens sensitive features of the SUT and only perturbs a single semantic feature at a time. To support a fair comparison, we primarily compare Mimicry against DETECT*, the simplified variant of our framework that does not use relevance attribution and perturbs all channels identified as sensitive. Mimicry serves as a strong baseline for evaluating generated test cases, and has been shown to surpass earlier methods such as Sinvad [30] and DeepJanus [55] in boundary testing settings.

SpRAY (spectral relevance analysis). An eigenvalue-based clustering method for spurious feature localization proposed by Lapuschkin et al. [5, 39]. It is based on computing Layer-wise Relevance Propagation (LRP) heatmaps for model predictions and then applying spectral clustering to group similar explanation patterns. We include SpRAY as a baseline to evaluate effectiveness in spurious feature detection, since it remains one of the few techniques that directly target the automatic detection of spurious features. SpRAY represents the pixel-level explanation methods and clusters saliency explanations. As such, SpRAY serves as a representative and methodologically relevant baseline for assessing the effectiveness of spurious feature detection from explanation patterns, complementing our feature-perturbation-based analysis.

5.4 Procedure

In RQ_1 , we apply each sensitive channel screening method using the confidence-based oracle on the facial task for 50 randomly selected seeds for tractability. This is justified as channel sensitivity mainly depends on StyleGAN’s internal structure rather than task semantics, making the results broadly transferable and reused across datasets in RQ_2 and RQ_3 . We fine-tuned the perturbation $\epsilon = 10$ to induce visible semantic changes and avoid artifacts, following existing work [44, 56]. Additionally, we analyze the contribution of specific synthesis layer channels to justify the selection of the number of candidate channels in subsequent experiments. To compute the human-aligned metrics for $RQ_{1,3}$, we collected human annotations using AWS Mechanical Turk [62]. Two tasks were evaluated: feature attribution and boundary-image relabeling. In total, 69 participants completed 1,069 annotation questions. We included several attention-check questions to detect inattentive responses and discarded 22 submissions that failed these checks.

In RQ_2 , we compare the best-performing configuration from RQ_1 with MIMICRY. For this comparison, we use the misclassification-based oracle, which resembles the boundary-based objectives used in MIMICRY [74] and evaluate with 50 seeds by a random number generator (RNG). Because DETECT independently perturbs hundreds of channels across all layers for each seed, this configuration yields 10,750 attempted test cases per model. Increasing the number of seeds would further improve statistical robustness in seed-level estimates, but the generation and annotation costs grow approximately linearly. Following existing guidelines in generative AI-based test generation [34, 44, 47], DETECT applies latent truncation to improve image quality. For the face and dogs tasks, we use a truncation value of $\psi = 0.7$, which is the commonly recommended default in the StyleGAN literature [34, 47]. For the COCO-Cars dataset, vehicles possess rigid geometric constraints that are prone to generative artifacts at high diversity. To reduce visible artifacts of the generated test cases, we use $\psi = 0.5$, which is an empirically selected, task-specific setting that reduced obvious geometric artifacts during visual inspection. However, lower ψ may reduce sample diversity. To efficiently navigate to the decision boundary of SUT, DETECT employs a constrained, adaptive hill-climbing search. To ensure efficient convergence, we employ a dynamic step size, initialized at $\epsilon = 5$ (to ensure efficient reduction of confidence at the beginning), which is adaptively halved when no valid improvement is found over successive iterations. The search terminates after at most 20 iterations or when the step size becomes negligibly small ($< 10^{-3}$), ensuring precise boundary test cases.

Beyond runtime, we assess the visual impact of manipulations using d_2 -distance (lower is better) and MS-SSIM [73] (higher indicates greater structural similarity).

In RQ_3 , we evaluate spurious feature detection using 50 RNG seeds for all 4 SUTs across all 3 tasks. We use a 40% confidence drop threshold to define influential features in the confidence-based oracle, $\tau = 0.4|y[t]|$, to capture impactful shifts in prediction without overreacting to minor fluctuations. To justify the selection of this threshold, we conduct a detailed sensitivity analysis—balancing noise filtration and spurious signal retention—with full quantitative results and visualizations provided in [Appendix C](#). We

re-implemented the SpRAy method based on the paper description [39], as no official code was provided. The procedure includes the following steps: First, we generate LRP relevance maps for positive samples in the test set. Then, we preprocess the maps by resizing and flattening them into uniform vectors and perform eigengap analysis on the k-nearest neighbor affinity graph to determine the optimal number of clusters. Finally, we apply spectral clustering [51] and visualize clusters via t-SNE [68]. This process serves as a baseline to assess the capability of DETECT to identify spurious correlations.

In RQ₄, we fine-tune the pretrained CelebA ResNet classifier using a small repaired dataset generated by DETECT. More specifically, the repair dataset mixes original CelebA training samples (80%) with generated counterfactuals (20%). This fixed ratio was selected empirically to retain substantial exposure to the original distribution while introducing counterfactual cases. Only the final classification layer is updated, while the backbone remains frozen. The model is optimized with a small learning rate ($\text{lr} = 2 \times 10^{-5}$), early stopping for at most 20 epochs, and a loss targeting only the finetuned attribute (eyeglasses), ensuring that the model adapts specifically to corrected feature-label associations. To ensure that evaluation is conducted on data unseen during fine-tuning, we create a held-out generated test set by reserving 20% of all VLM-reabeled counterfactuals using a fixed random seed.

5.5 Results

5.5.1 RQ₁ (configuration). Table 1 summarizes the average confidence drop and runtime for each sensitive channel screening method. We used two-sided paired Wilcoxon signed-rank tests [76] on seed-level aggregated results with Holm correction [25] (p -values below 0.05 are considered statistically significant). We report matched-pairs rank-biserial correlations (r_{rb}) as the corresponding effect size, with absolute values of 0.1, 0.3, and 0.5 interpreted as small, medium, and large effects, respectively [35]. The two white-box methods, *Grad* and *SmoothGrad*, outperform the *Random* baseline in reducing model confidence while *FDA* does not differ significantly from random selection. In terms of runtime, *Grad*, *SmoothGrad*, and *Random* are efficient and perform similarly (~ 0.79 seconds). This indicates that the white-box screening methods incur near-zero computational overhead compared to random channel selection. *FDA*, by contrast, is slower due to its gradient-free sensitivity estimation.

To better understand how many sensitive channels are required for effective perturbation, we analyze the distribution of activated channels across StyleGAN's synthesis layers in cases where the SUT exhibits a confidence drop greater than 1. This threshold removes low-signal noise and retains only semantically impactful perturbations, as it reflects a substantial logit change.

As shown in Fig. 6, only a subset of channels, mainly from early layers, are consistently involved in these impactful cases. This analysis suggests that a limited number of candidate channels is sufficient to induce meaningful behavioural shifts while maintaining efficiency. Specifically, we select the top-15 most sensitive channels from the coarse-to-middle layers, and the top-5 from the fine layers (beyond 512 resolution) for subsequent tasks. While earlier layers are known to control high-level semantics [33],

Table 1. RQ₁ (configuration) results and statistical inference.

	<i>Grad</i>	<i>SmoothGrad</i>	<i>FDA</i>	<i>Random</i>
Confidence Drop $\uparrow$	0.370 $\pm$ 0.381	0.470 $\pm$ 0.416	0.052 $\pm$ 0.156	0.048 $\pm$ 0.161
Runtime (sec) $\downarrow$	0.786	0.789	1.577	0.797
<i>p</i> -values	5.33×10^{-15} ★	5.33×10^{-15} ★	0.826	–
Δ [95% CI]	0.322 [0.241, 0.430]	0.422 [0.332, 0.533]	0.0049 [–0.0056, 0.0166]	–
r_{tb}	1.000 ★	1.000 ★	0.037	–

Note. **Bold** values indicate the best results, and ★ indicates significant differences ($p < 0.05$) or large effects ($|r_{tb}| \geq 0.5$). *p*-values are obtained from two-sided paired Wilcoxon signed-rank tests with Holm correction over the three comparisons; Δ is the paired mean difference, and CI denotes its 95% bootstrap confidence interval; r_{tb} denotes matched-pairs rank-biserial correlation as effect size.

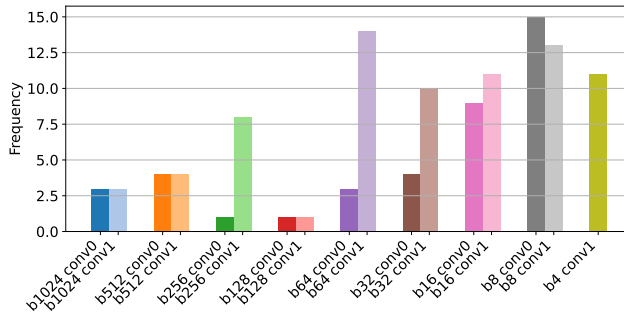


Fig. 6. RQ₁ (configuration): Number of channels with confidence drop greater than one for each StyleGAN layer in DETECT.

our results confirm that their channels are disproportionately represented among those driving significant output changes.

Fig. 7 compares the performance of 7 VLMs across four dimensions. The top row shows their agreement with human annotators for the two tasks: Relevance Attribution and Relabeling. Each bar pair reports both the raw agreement and the majority-vote consensus agreement. The Relevance Attribution task is more challenging; models such as qwen3-v1-8b-thinking and gpt-o4-mini achieve the highest alignment with human judgments. For the Relabeling task, performance is generally more stable: all models except mistral-small-3.1-24b-instruct attain high agreement levels close to the human upper bound. The bottom row presents the cost per query and generation time, revealing substantial differences in efficiency: lighter models incur minimal cost and latency, while larger models such as GPT-4.1 exhibit noticeably higher computational overhead. Considering both semantic alignment and efficiency trade-offs in cost and latency, we select qwen3-v1-8b-thinking as the VLM used in all subsequent experiments.

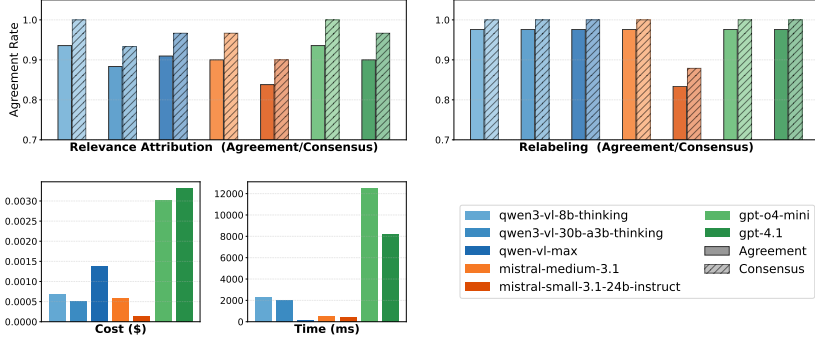


Fig. 7. RQ_1 (configuration): Performance comparison of the evaluated VLMs.

RQ_1 (configuration): SmoothGrad offers the best trade-off between attribution quality and computational cost among sensitivity screening methods. Analysis of impactful perturbations reveals that selecting top-15 channels per early/middle layer and top-5 per fine layer balances effectiveness with efficiency and captures most impactful perturbations. Across both attribution and relabeling tasks, the evaluated VLMs show high alignment with human judgments, with qwen3-vl-8b-thinking providing the best balance of semantic accuracy and efficiency.

5.5.2 RQ_2 (generalization via behaviour exploration). Table 2 shows the result of generalization of DETECT and DETECT* compared with MIMICRY. For COCO-Cars and Dogs we report only DETECT*, since the full pipeline evaluates a relevance-filtered subset that does not admit a like-for-like paired comparison with Mimicry; full DETECT is therefore shown only for the two CelebA tasks. Across all four evaluated tasks, DETECT* generates more structurally similar boundary cases with smaller image changes and substantially lower runtime than Mimicry. Boundary-distance performance is task-dependent: DETECT* performs better on CelebA, Mimicry performs better on COCO-Cars, and neither method has a statistically reliable advantage on Dogs. These inferential conclusions concern DETECT*, as the full DETECT performs semantic relevance filtering and therefore evaluates a task-relevant subset that does not provide a like-for-like paired comparison.

We also provide visual comparison of the generated examples for DETECT and MIMICRY across all four tasks in Appendix A. These examples illustrate the qualitative differences between the two methods and support the quantitative trends reported above. In particular, DETECT generally produces more focused and attribute-specific edits, while MIMICRY often introduces broader or incidental changes.

Table 2. RQ₂: Effectiveness and efficiency results (**best**; ★ statistically significant or large effect).

		Runtime (sec) ↓	$\tilde{d}_2$ -Image ↓	MS-SSIM ↑	d_2 -Boundary ↓
CelebA glasses	DETECT	1.288 ± 0.354	0.084 ± 0.025	0.730 ± 0.054	0.035 ± 0.051
	DETECT*	1.256 ± 0.384	0.088 ± 0.043	0.731 ± 0.053	0.060 ± 0.141
	MIMICRY	96.795 ± 0.751	0.250 ± 0.049	0.420 ± 0.073	2.996 ± 0.434
	p -value	1.91×10 ⁻⁵ ★	1.91×10 ⁻⁵ ★	1.91×10 ⁻⁵ ★	1.91×10 ⁻⁵ ★
	r_{fb}	1.000 ★	1.000 ★	1.000 ★	1.000 ★
	Δ (95% CI)	-95.410 [-95.582, -95.157]	-0.176 [-0.198, -0.141]	0.317 [0.277, 0.363]	-2.955 [-3.087, -2.794]
CelebA gender	DETECT	1.277 ± 0.354	0.098 ± 0.025	0.668 ± 0.090	2.442 ± 2.275
	DETECT*	3.846 ± 2.830	0.115 ± 0.083	0.714 ± 0.087	0.566 ± 1.109
	MIMICRY	96.342 ± 0.676	0.247 ± 0.046	0.415 ± 0.065	1.867 ± 0.624
	p -value	1.48×10 ⁻¹² ★	4.46×10 ⁻⁸ ★	1.48×10 ⁻¹² ★	2.88×10 ⁻⁴ ★
	r_{fb}	1.000 ★	0.897 ★	1.000 ★	0.671 ★
	Δ (95% CI)	-94.974 [-95.101, -94.874]	-0.133 [-0.149, -0.095]	0.234 [0.188, 0.290]	-0.983 [-1.180, -0.691]
COCO Cars	DETECT*	3.247 ± 2.208	0.094 ± 0.045	0.759 ± 0.072	0.644 ± 0.038
	MIMICRY	50.982 ± 0.702	0.251 ± 0.044	0.332 ± 0.083	0.259 ± 0.200
	p -value	3.05×10 ⁻⁴ ★	3.05×10 ⁻⁴ ★	3.05×10 ⁻⁴ ★	3.05×10 ⁻⁴ ★
	r_{fb}	1.000 ★	1.000 ★	1.000 ★	-0.983 ★
	Δ (95% CI)	-49.917 [-49.968, -48.550]	-0.166 [-0.182, -0.137]	0.502 [0.464, 0.575]	0.525 [0.397, 0.619]
Dogs	DETECT*	3.697 ± 2.423	0.073 ± 0.043	0.819 ± 0.100	0.457 ± 0.118
	MIMICRY	12.325 ± 0.674	0.281 ± 0.065	0.157 ± 0.134	0.422 ± 0.157
	p -value	2.84×10 ⁻¹⁴ ★	5.33×10 ⁻¹⁴ ★	5.33×10 ⁻¹⁴ ★	0.399
	r_{fb}	1.000 ★	0.998 ★	0.998 ★	-0.139
	Δ (95% CI)	-10.011 [-10.584, -9.380]	-0.211 [-0.222, -0.192]	0.651 [0.616, 0.693]	-0.010 [-0.051, 0.074]

Note. **Bold** indicates the best result for each task and metric. ★ indicates significant differences ($p < 0.05$) or large effects ($|r_{fb}| \geq 0.5$). The statistical rows compare DETECT* with MIMICRY using Holm-adjusted p -values from two-sided paired Wilcoxon signed-rank tests on seed-level aggregates; r_{fb} denotes matched-pairs rank-biserial correlation as effect size; Δ is the paired mean difference between DETECT* and MIMICRY, and CI denotes its 95% bootstrap confidence interval.

Table 3. RQ₃: Comparison of spurious feature detection across ResNet-Glasses, SWAG-Glasses, and ReXNet-Dog. Only frequently used channels (by > 1 input) are retained.

	ResNet-Glasses	SWAG-Glasses	ReXNet-Dog
Influential inputs (selected/ total)	415/548	168/524	2294/3784
Channels used (selected/total)	104/236	95/450	621/1321
Relevant/Spurious inputs	335/80	55/113	1/2293
Relevant/Spurious channels	71/33	39/56	0/621
$R_{Relevance}$	0.68	0.41	0.00

RQ₂ (generalization via behaviour exploration): DETECT* produces more targeted and visually precise manipulations than MIMICRY while preserving structural similarity at a lower runtime, confirming more controlled and efficient test generation.

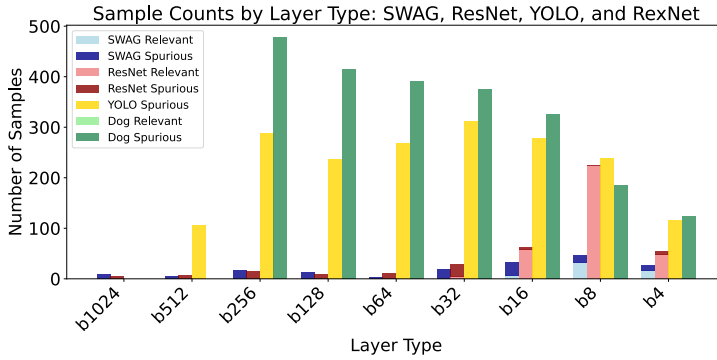


Fig. 8. RQ₃: distribution of relevant and spurious attributions across StyleGAN layers for SWAG, ResNet, YOLO, and ReXNet.

5.5.3 RQ₃ (robustness via spurious feature detection). Table 3 reports the results for spurious feature identification in glasses classification, separately for a fully fine-tuned ResNet50 and a frozen-backbone SWAG ViT. Although both models achieve high test accuracy on the eyeglasses attribute, their behaviour under controlled latent perturbations diverges substantially.

ResNet50 identifies more task-relevant influential inputs (335 vs. 55) and channels (71 vs. 39), whereas SWAG yields more spurious influential inputs (113 vs. 80) and channels (56 vs. 33). The relevance ratio $R_{\text{Relevance}}$ is higher for ResNet (0.68 vs. 0.41), indicating stronger alignment with semantically meaningful features. Despite SWAG’s theoretical robustness [46], it shows greater reliance on spurious features. These results suggest that SWAG’s frozen backbone may limit task adaptation, increasing dependence on incidental cues. ReXNet-Dog exhibits a markedly different attribution profile. Across the retained inputs, only one was labeled task-relevant, and no relevant channel remained after the frequency-based channel aggregation. Both statistics from RQ₂ (Table 2) and the concrete examples in Appendix A (Fig. 11) consistently support the interpretation that very small image-level appearance changes can induce ReXNet’s failures without visibly altering the breed identity, indicating that ReXNet is inadequate for this fine-grained classification task. That said, because the Dog Breed attributions rely on a general-purpose VLM without expert breed-level validation, and because ReXNet’s lower task accuracy makes shortcut attribution inherently noisier, we treat the Dog Breed results as an exploratory case study rather than a full comparison, and correspondingly restrict our cross-dataset effectiveness claims to the CelebA and COCO-Cars settings.

To analyze these results more qualitatively, Fig. 8 shows the distribution of relevant and spurious features across StyleGAN layers. Relevant channels concentrate in coarse blocks (b4–b16), which encode semantic structures like shape. In contrast, spurious channels appear more often in mid-to-late layers (e.g., b512 and above), linked to texture and lighting, confirming that many spurious dependencies stem from low-level

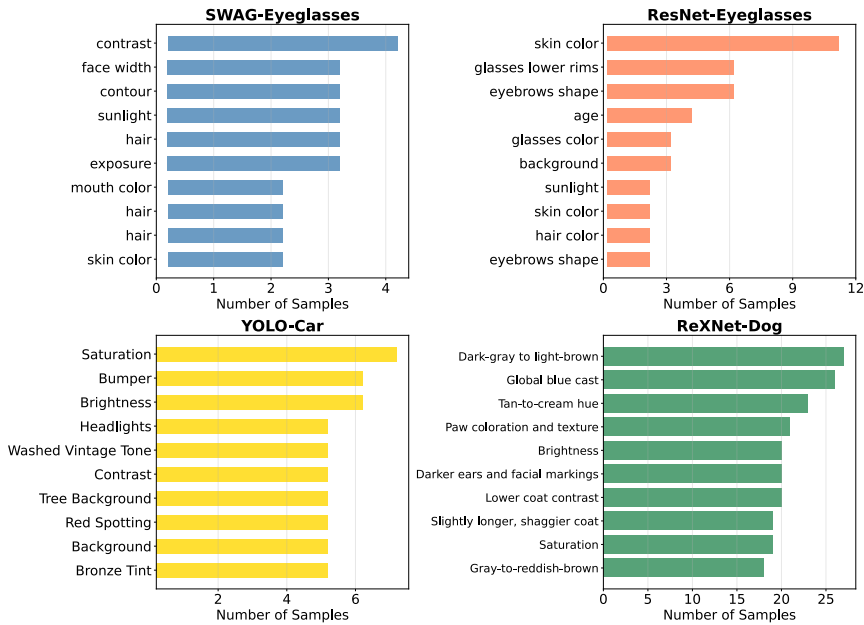


Fig. 9. RQ3: Top-10 spurious features identified for SWAG, ResNet, YOLO, and ReXNet

artifacts. While ResNet focuses on relevant channels, SWAG exhibits a more dispersed activation pattern with fewer relevant and more spurious attributions, reinforcing the risk of lacking adaptation to relevant features in a weakly supervised manner. Both models' spurious activations concentrate in mid-to-late layers tied to lighting and background, underlining the value of layer-aware attribution for diagnostic insights. For YOLO-based car detection, the spurious attributions are spread more uniformly across all layers, unlike the eyeglasses task where they are confined to coarse-level features. For the dog breed task, with the same number of seeds, ReXNet was affected by the largest number of samples, which were concentrated in the mid-to-late layers.

Fig. 9 further reveals that SWAG's top spurious directions often reflect global appearance cues, such as face width, contour, lighting intensity, and exposure. ResNet, on the other hand, has semantically finer spurious directions, such as glasses frame style, eyebrow height, or clothing. This contrast suggests that ResNet latches onto plausible but misleading features, while SWAG responds more to structural or environmental noise. This finding echoes recent work on ViTs [59], which demonstrates that global attention mechanisms often degenerate into utilizing background patches as representation shortcuts. For YOLO car detection, the spurious features span a broader range, including shifts in color saturation, bronze tint, and contrast, indicating that the car detector may rely on a wider mix of context signals than the attribute classifiers. For fine-grained Dog Breed classification, the top spurious features for the dog classifier are overwhelmingly dominated by environmental factors (e.g., global

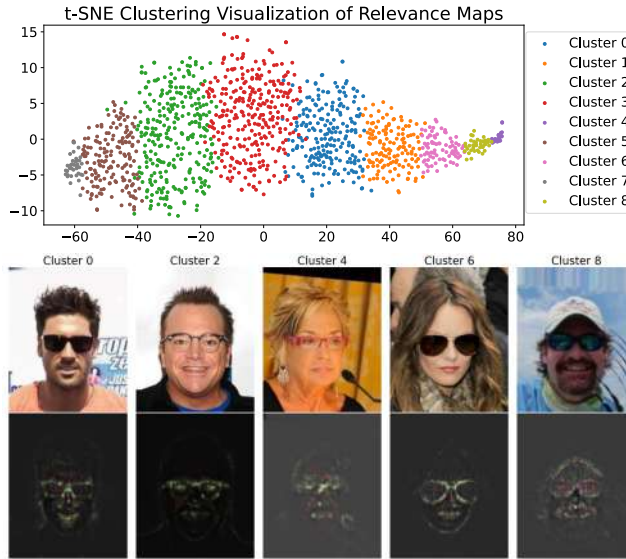


Fig. 10. RQ₃: t-SNE visualization of relevance maps by SpRAY. Bottom: samples from five distinct clusters, including original images and corresponding relevance maps.

color cast, brightness) and generic localized features (e.g., fur color, texture) rather than distinct morphological breed traits.

Finally, Fig. 10 shows a t-SNE of relevance maps clustered using SpRAY. While the eigengap analysis suggests 9 clusters, the visualization shows a mostly continuous distribution. Cluster inspection reveals repeated attention around eyeglasses, raising ambiguity over whether these activations reflect causal semantics or spurious correlations. This highlights the limitations of pixel-level relevance methods and the strength of our feature-space probing in uncovering subtle spurious patterns.

RQ₃ (robustness via spurious feature detection): *DETECT reveals substantial differences in spurious feature reliance between models with similar accuracy, showing that SWAG ViT focuses more on low-level, global cues while ResNet attends to finer but still irrelevant traits. In contrast, SpRAY does not reveal equally spurious patterns in the evaluated setting, especially when irrelevant features co-occur with class semantics.*

5.5.4 RQ₄ (model repair). On the CelebA test set, the fine-tuned model maintains performance comparable to the original model (target accuracy remains 0.997), indicating that tuning on repaired samples does not degrade performance on standard data. On the generated boundary-level test set, the improvements are clear: target accuracy increases from 0.935 to 1.000. These results show that the repaired dataset helps the model handle relevant feature perturbations more reliably while preserving its original behaviour.

RQ₄ (model repair): *We provide evidence that DETECT-generated counterfactual tests can support model repair. The fine-tuning process successfully boosted accuracy on the generated test set from 0.935 to 1.000, while preserving the standard test accuracy.*

5.6 Threats to Validity

5.6.1 Internal validity. The performance of DETECT depends on the quality of the pre-trained StyleGAN generator and the accuracy of the VLMs used in feature attribution. Errors in feature relevance estimation or latent disentanglement could lead to incorrect categorization of features as task-relevant or spurious. To mitigate this, we rely on pre-trained models when possible, and channel labels are aggregated across multiple counterfactual samples. Additionally, the threshold selection for sensitivity and logit change might affect the precision of behavioural attribution.

Furthermore, DETECT relies on a VLM to construct the feature-aware test oracle. A potential threat to construct validity is the intrinsic bias of the chosen VLM. We mitigated this threat by carefully selecting state-of-the-art VLMs that show high alignment with human visual instruction following. Furthermore, we empirically assessed this threat through a human study (RQ_{1.3}), confirming that the VLM’s attribution and relabeling decisions strongly align with human consensus.

5.6.2 External validity. Our evaluation is limited to specific datasets (*CelebA*, *COCO-Cars*, *Dogs*) and tasks (classification and detection). While we fine-tuned StyleGAN models for better alignment, results may not generalize to domains with less disentangled latent spaces or less structured visual features. Furthermore, although we evaluate DETECT on three distinct model architecture types, the results may not generalize to all model families or training regimes. In particular, models with different characteristics—such as parameter sizes, architectural designs, or hyperparameter configurations—may exhibit different behaviours when applying our approach.

5.6.3 Reproducibility. All software artifacts and the complete results set are available in our replication package [1].

6 DISCUSSION

6.1 From Class- to Feature-Level Testing

Most prior DL testing methods lack semantic control, making test failures hard to interpret [30, 55]. MIMICRY introduced class-conditional testing, improving usability but still limited by the coarser $\mathcal{W}$ -space. In contrast, DETECT operates in $\mathcal{S}$ -space and targets specific semantic features based on model behaviour, enabling fine-grained objectives (as presented in Fig. 11). DETECT currently perturbs single feature dimensions at a time, which is effective for localized attributes (e.g., glasses) but may be less optimal for composite global concepts (e.g., gender, car). Nevertheless, DETECT by fine-grained control over a single feature still outperforms the state-of-the-art baseline on both efficiency and image preservation. As demonstrated in RQ₂ (Table 2), DETECT* significantly reduces runtime and achieves higher image preservation than MIMICRY, while maintaining comparable boundary distances. Future work will explore

simultaneous perturbations of multiple correlated features to further enhance the manipulation of complex relevant attributes.

6.2 Spurious Detection vs. Classifier Quality

The utility of spurious feature detection is closely tied to classifier quality. For low-performing models, such as in fine-grained multi-class dog breed classification, misclassifications frequently stem from underfitting or class confusion rather than shortcut reliance, making spurious attribution less meaningful. For these models, improving their basic robustness may be the most immediate priority. DETECT* remains useful for exposing this instability, whereas fine-grained relevance aggregation may provide limited additional structure when almost no recurring relevant channel can be identified.

Conversely, for high-performing classifiers, accuracy is not sufficient to assess reliability. As shown in RQ₃ (Table 3 and Fig. 8), both ResNet and SWAG achieve strong accuracy on the glasses task, yet our analysis reveals they rely on different features, including quite distinct spurious patterns. These findings reinforce the need for feature-aware testing that distinguishes causal semantics from confounding correlations, offering diagnostic insights that go beyond accuracy metrics. To this aim, the proposed relevance ratio provides a complementary perspective to conventional accuracy.

6.3 Explainability in Test Generation

Incorporating feature-aware perturbations provides a structured form of explainability. Rather than blindly searching for misclassification like MIMICRY, DETECT distinguishes between task-relevant and irrelevant features, enabling a more informative testing process and more precise judgments about model behaviour: failures triggered by irrelevant features reveal spurious reliance (i.e., lack of robustness), while those triggered by relevant ones indicate a lack of generalization.

This fine-grained analysis is particularly important for the SUTs, which achieve high accuracy on the test set. We complement it with additional insights on *why* they fail on the features identified by DETECT, as demonstrated in RQ₃ (Fig. 9). Our framework supports this by converting test generation into a targeted probe of the model's feature dependence. This not only improves test quality but also enables practical feedback for model assessment, debugging, and retraining.

6.4 Latent-Space Interpretability

Compared to conventional saliency-based methods, our framework introduces a more structured form of explainability grounded in latent space semantics. As shown in RQ₃ (Fig. 10), methods like SpRAy produce pixel-level heatmaps that highlight regions on which the model focuses, but offer limited insight into what types of features affect the prediction. In contrast, DETECT explains model behaviour in terms of interpretable latent factors, offering concept-level attribution rather than low-level localization. DETECT bridges the gap between interpretability and explainability: it does not merely visualize what the model reacts to, but enables structured reasoning about why it reacts that way.

Table 4. Comparison of DETECT with existing DL evaluation paradigms across guidance, operating spaces, and oracles.

Methodology (Representative Works)	Guidance Space (What directs the search?)	Operating Space (Where are changes applied?)	Semantic Interpretability	Automated Semantic Oracle
Adversarial Testing (e.g., Szegedy et al. [65])	SUT Gradients / Loss (Mathematical)	Raw Pixel Space (Input Dimensions)	×	Original Label (via Proximity)
Coverage & Fuzzing (e.g., DeepXplore [53], DSA [36])	Internal SUT Activations (Neural Representations)	Raw Pixel Space (Adding Noise)	×	Differential / Proximity
Macro Generative Testing (e.g., DeepRoad [81])	Predefined Metamorphic Relations (Rules)	Image-to-Image (Macro Styles)	~ (Holistic)	~ (Rigid MRs)
Latent Boundary Search (e.g., MIMICRY [74])	SUT Confidence (Prediction Shifts)	Latent $\mathcal{W}$ -Space (Composite Vectors)	× (Entangled)	Original Label (via Proximity)
Feature-Aware Testing DETECT (Ours)	Latent $\mathcal{S}$-Space (Feature Sensitivity)	Latent $\mathcal{S}$-Space (Disentangled Channels)	✓ (Disentangled + VLM)	✓ (VLM-based)

Note: ✓ denotes fully supported, ~ denotes partial support or limitations, and × denotes not supported. MRs = Metamorphic Relations.

7 RELATED WORK

7.1 Test Generation for DNNs

There are several strategies for generating test inputs for deep learning systems. To clarify the fundamental novelty of DETECT, we contrast it with existing paradigms across their guidance mechanisms, operating spaces, and automated test oracles (as illustrated in Table 4).

Raw Input Manipulation. These methods work directly on the input space, often by perturbing pixel values to induce misbehaviours [38]. Early adversarial methods generate test cases by directly applying mathematical perturbations to the raw input space to maximize prediction errors [65]. To provide better search direction, a prominent line of subsequent research proposed sampling the internal feature space of the SUT. This includes structural criteria like neuron coverage [20, 43, 53], Distance-based Surprise Adequacy (DSA) [36], and representation-guided fuzzing (e.g., TensorFuzz) [52]. A recent empirical study [10] extends attribution-guided testing to task-specialized language models, combining token-level attribution with LLM-driven, semantics-preserving mutation and automated validity checks. Beyond test generation, attribution signals have also been used for runtime monitoring of DL systems [19, 64], e.g., turning attention maps into confidence scores to anticipate misbehaviours [63].

DETECT differs from these methods in three fundamental aspects. First, in terms of the guidance space, their search is directed by traversing the SUT’s internal activation states. However, SUT activations are entangled vectors optimized solely for a specific discriminative task. In contrast, DETECT is guided directly within the disentangled and more generalized Latent $\mathcal{S}$ -space by identifying sensitive semantic channels. Second, regarding the operating space, these raw input manipulation methods apply noise or image translation. While effective, they often produce unrealistic artifacts and are limited in exploring novel, semantically meaningful input variations [56]. Ultimately, exploring the uninterpretable activations in SUT and pixel-level modifications prevents them from synthesizing clean, feature-level counterfactuals. Third, concerning the test oracle, these test generation methods predominantly reuse the original input’s label. They rely on the restrictive assumption that by enforcing small

perturbation magnitudes or applying similarity constraints, the underlying semantics are preserved. However, this proximity-based oracle often fails to guarantee true semantic correctness [56].

Model-based Techniques. They generate test inputs by manipulating high-level, structured representations of the input domain, such as parameterized models encoding domain constraints [15, 50, 55]. These methods enable the synthesis of semantically meaningful and novel inputs by altering the model and converting it back to raw inputs. However, they rely on manually crafted domain models, which can limit their generalizability [56].

Generative AI-based Techniques. They have recently gained attention for their ability to produce structurally novel test cases without domain models. Seminal works in this domain, such as DeepRoad [81], utilize GANs for metamorphic testing by applying macroscopic image-to-image translations (e.g., altering driving scenes from sunny to snowy). While effective for domain-level robustness, they lack fine-grained control over individual objects. While Variational Autoencoders (VAEs) have been used [14, 30, 44], they struggle with complex, context-rich image data [74]. GANs offer improved generative power [44, 74, 81], and recent work is beginning to explore diffusion models as well [8, 16, 44, 49, 75]. Yet, they usually explore the latent space without distinguishing semantic features. The search in latent space often follows directions that mix attributes or rely on interpolation paths that drift toward invalid regions of the latent manifold [30]. Furthermore, boundary-search methods in latent spaces still largely rely on original labels via similarity preservation, which remains prone to semantic drift.

DETECT uses a generative AI-based technique, leveraging StyleGAN’s architecture for fine-grained control over image features—a capability not yet explored in prior work. It integrates spurious feature detection into the generation process to increase the effectiveness of test cases. DETECT advances DL testing by aligning its guidance, manipulation, and oracle mechanisms around explicitly disentangled semantics. Instead of uninterpretable SUT activations or entangled latent spaces, DETECT uses the S -space both to guide its search and to apply perturbations. This allows the search to identify and perturb sensitive features via precise 1D channel-wise semantic traversals. Instead of assuming label preservation through rigid similarity constraints, DETECT incorporates a VLM-based oracle. This actively evaluates whether the generated counterfactuals correctly reflect the expected human-understandable concepts, mapping SUT failures to semantic vulnerabilities rather than mere pixel variations.

7.2 Spurious Correlation Detection/Mitigation

Feature invariance is a foundational principle in robust and fair machine learning, dictating that a model’s output should remain stable under variations in input features that are semantically irrelevant to the target task. To highlight the unique positioning of DETECT, we categorize existing literature into three paradigms and contrast them with our approach.

Training-Time Invariance and Mitigation. The vast majority of prior work addresses spurious correlations by enforcing invariance actively during the model training phase. Prominent strategies include Invariant Risk Minimization (IRM) [6], bias-mitigation losses [4, 72], gradient-based debiasing [13], and environment-aware optimization [27]. Furthermore, architectural modifications and specific pretraining designs (e.g., robust ViT variants) have been proposed to naturally resist spurious features [18, 46]. Overall, these methods require full access to the training information or the ability to modify the objective function. In contrast, DETECT targets post-hoc testing. It is designed to evaluate, test, and diagnose already-deployed models.

Data-Driven Interventions and Counterfactuals. Another line of work focuses on mitigating bias via dataset interventions. Methods in fairness-aware learning propose counterfactual tests [37], while vision-specific approaches manually alter image backgrounds (e.g., in animal classification [9]) or apply masking to irrelevant regions [67]. However, collecting naturally occurring counterfactual datasets or manually annotating bounding boxes to decouple features is notoriously labor-intensive and struggles to scale effectively in complex visual domains [3]. DETECT overcomes this data bottleneck. By leveraging the disentangled Latent $\mathcal{S}$ -space, DETECT *automatically* synthesizes high-fidelity, fine-grained counterfactuals (e.g., changing only the background or lighting), eliminating the reliance on manually curated datasets or annotations.

Post-Hoc Spurious Detection. Less work has been directed toward testing trained DL models for faulty behaviour. Existing post-hoc detection methods often rely on injecting artificial shortcut signals (e.g., manually placing watermarks or color patches) into test sets to observe model reliance [5, 39], or they utilize XAI heatmaps to highlight activated regions. However, these methods still require manual annotation on the highlighted regions. Moreover, XAI heatmaps suffer from pixel-level entanglement and cannot generate executable test cases. DETECT differs by actively perturbing natural, real-world semantics.

DETECT shifts the focus from training-time mitigation and pixel-level post-hoc explanation to an automated, generative testing framework. While existing methods can spatially separate objects from backgrounds, they fail to untangle conceptual features that naturally overlap in the pixel space (e.g., lighting, skin color, eyebrows). DETECT overcomes this by projecting saliency analysis directly into the disentangled $\mathcal{S}$ -space. This allows us to actively and automatically decouple fine-grained semantics and perturb them individually to confirm invariance violations. Moreover, by employing a VLM as an automated semantic oracle, DETECT automatically assesses whether the SUT's prediction shifts are legitimately caused by task-relevant feature changes or are the direct result of spurious feature reliance. This provides an end-to-end testing framework for exposing shortcut learning in complex vision models.

8 CONCLUSIONS

We presented DETECT, a feature-aware test generation framework based on test generation and spurious feature diagnosis. By operating in a semantically disentangled latent space, DETECT enables fine-grained perturbations aligned with task semantics, and distinguishes between relevant and spurious feature influences. Our experiments

show that DETECT uncovers behaviourally significant vulnerabilities in high-accuracy models, providing insight into how different models rely on distinct types of features.

As part of our future work, we plan to extend the framework beyond StyleGAN to other generative architectures, such as diffusion-based models and transformer-based generators. While GANs can exhibit relatively structured and partially disentangled latent spaces, diffusion models currently represent the state of the art in generative modeling but pose additional challenges for feature-aware testing. In particular, their latent representations are iteratively denoised rather than decoded in a single step, making feature attribution and controlled manipulation less straightforward than in architectures with explicit intermediate synthesis layers. Transformer-based generators pose related challenges, as semantic features may be distributed across multiple patches and channels, making controlled perturbations more difficult. Another direction is extending the approach to modalities outside vision, such as natural language or audio, where latent structure and semantic drift present related challenges. These directions could broaden the applicability of DETECT to a wider range of DL testing.

REFERENCES

- [1] 2026. Replication Package. <https://github.com/ast-fortiss-tum/DETECT>.
- [2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).
- [3] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. 2021. Invariance principle meets information bottleneck for out-of-distribution generalization. In *Proceedings of the 35th International Conference on Neural Information Processing Systems (NIPS '21)*. Curran Associates Inc., Red Hook, NY, USA, Article 263, 13 pages.
- [4] Mohsan Alvi, Andrew Zisserman, and Christoffer Nellåker. 2018. Turning a Blind Eye: Explicit Removal of Biases and Variation from Deep Neural Network Embeddings. In *Computer Vision – ECCV 2018 Workshops: Munich, Germany, September 8-14, 2018, Proceedings, Part I* (Munich, Germany). Springer-Verlag, Berlin, Heidelberg, 556–572. https://doi.org/10.1007/978-3-030-11009-3_34
- [5] Christopher J Anders, Leander Weber, David Neumann, Wojciech Samek, Klaus-Robert Müller, and Sebastian Lapuschkin. 2022. Finding and removing clever hans: Using explanation methods to debug and improve deep models. *Information Fusion* 77 (2022), 261–295.
- [6] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893* (2019).
- [7] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS one* 10, 7 (2015), e0130140.
- [8] Luciano Baresi, Davide Yi Xian Hu, Andrea Stocco, and Paolo Tonella. 2025. Efficient Domain Augmentation for Autonomous Driving Testing Using Diffusion Models. In *Proceedings of the 47th International Conference on Software Engineering (ICSE '25)*. IEEE, 12 pages.
- [9] Sara Beery, Grant Van Horn, and Pietro Perona. 2018. Recognition in Terra Incognita. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI* (Munich, Germany). Springer-Verlag, Berlin, Heidelberg, 472–489. https://doi.org/10.1007/978-3-030-01270-0_28
- [10] Xingcheng Chen, Mehmet Besenk, and Andrea Stocco. 2026. Explanation-Guided Metamorphic Testing of Specialized Language Models: An Empirical Study. In *Proceedings of the 2026 International Symposium on Empirical Software Engineering and Measurement (ESEM '26)*. IEEE. <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ESEM.2026.5>

- [11] Xingcheng Chen, Matteo Biagiola, Vincenzo Riccio, Marcelo d'Amorim, and Andrea Stocco. 2026. XMutant: XAI-based Fuzzing for Deep Learning Systems. *Empirical Software Engineering* (2026). <https://doi.org/10.1007/s10664-025-10792-1>
- [12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. <https://doi.org/10.1109/CVPR.2009.5206848>
- [13] Yihe Deng, Yu Yang, Baharan Mirzasoleiman, and Quanquan Gu. 2023. Robust learning with progressive data expansion against spurious correlation. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) (NIPS '23). Curran Associates Inc., Red Hook, NY, USA, Article 69, 13 pages.
- [14] Swaroopa Dola, Rory McDaniel, Matthew B Dwyer, and Mary Lou Soffa. 2024. CIT4DNN: Generating Diverse and Rare Inputs for Neural Networks Using Latent Space Combinatorial Testing. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–13.
- [15] Alessio Gambi, Marc Mueller, and Gordon Fraser. 2019. Automatically Testing Self-driving Cars with Search-based Procedural Content Generation. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis* (Beijing, China) (ISSTA 2019). ACM, New York, NY, USA, 318–328. <https://doi.org/10.1145/3293882.3330566>
- [16] Yuan Gao, Mattia Piccinini, Yuchen Zhang, Dingrui Wang, Korbinian Moller, Roberto Brusnicki, Baha Zarrouki, Alessio Gambi, Jan Frederik Totz, Kai Storms, Steven Peters, Andrea Stocco, Bassam Alrifae, Marco Pavone, and Johannes Betz. 2026. Foundation Models in Autonomous Driving: A Survey on Scenario Generation and Scenario Analysis. *IEEE Open Journal of Intelligent Transportation Systems* (2026). <https://arxiv.org/abs/2506.11526>
- [17] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence* 2, 11 (2020), 665–673.
- [18] Soumya Ghosal and Yixuan Li. 2023. Are Vision Transformers Robust to Spurious Correlations? *International Journal of Computer Vision* 132 (10 2023), 1–21. <https://doi.org/10.1007/s11263-023-01916-5>
- [19] Ruben Grewal, Paolo Tonella, and Andrea Stocco. 2024. Predicting Safety Misbehaviours in Autonomous Driving Systems using Uncertainty Quantification. In *Proceedings of the 17th IEEE International Conference on Software Testing, Verification and Validation (ICST '24)*. IEEE, 12 pages.
- [20] Jianmin Guo, Yu Jiang, Yue Zhao, Quan Chen, and Jianguang Sun. 2018. Dlfuzz: Differential fuzzing testing of deep learning systems. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 739–743.
- [21] Dongyoon Han, Sangdoo Yun, Byeongho Heo, and YoungJoon Yoo. 2021. Rethinking Channel Dimensions for Efficient Model Design. *arXiv:2007.00992* [cs.CV]
- [22] Yujin Han and Difan Zou. 2024. Improving group robustness on spurious correlation requires preciser group inference. In *Proceedings of the 41st International Conference on Machine Learning* (Vienna, Austria) (ICML '24). JMLR.org, Article 696, 25 pages.
- [23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
- [24] Zhenliang He, Meina Kan, and Shiguang Shan. 2021. EigenGAN: Layer-Wise Eigen-Learning for GANs. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 14388–14397. <https://doi.org/10.1109/ICCV48922.2021.01414>
- [25] Sture Holm. 1979. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics* 6, 2 (1979), 65–70.
- [26] Nargiz Humbatova, Gunel Jahangirova, Gabriele Bavota, Vincenzo Riccio, Andrea Stocco, and Paolo Tonella. 2020. Taxonomy of Real Faults in Deep Learning Systems. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE '20)*. ACM, 12 pages. <https://doi.org/10.1145/3377811.3380395>
- [27] Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew Gordon Wilson. 2022. On feature learning in the presence of spurious correlations. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) (NIPS '22). Curran Associates Inc., Red Hook,

- NY, USA, Article 2791, 17 pages.
- [28] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] <https://arxiv.org/abs/2310.06825>
 - [29] Glenn Jocher, Jing Qiu, and Ayush Chaurasia. 2023. Ultralytics YOLO. <https://ultralytics.com> Released under AGPL-3.0. Repository: <https://github.com/ultralytics/ultralytics>.
 - [30] Sungmin Kang, Robert Feldt, and Shin Yoo. 2020. Sinvad: Search-based image space navigation for dnn image classifier test input generation. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*. 521–528.
 - [31] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2020. Training generative adversarial networks with limited data. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (Vancouver, BC, Canada) (NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 1015, 11 pages.
 - [32] Tero Karras, Miika Aittala, Samuli Laine, Erik H  rk  nen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2021. Alias-Free Generative Adversarial Networks. In *Proc. NeurIPS*.
 - [33] Tero Karras, Samuli Laine, and Timo Aila. 2021. A Style-Based Generator Architecture for Generative Adversarial Networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 43, 12 (Dec. 2021), 4217–4228. <https://doi.org/10.1109/TPAMI.2020.2970919>
 - [34] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and Improving the Image Quality of StyleGAN. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 8107–8116. <https://doi.org/10.1109/CVPR42600.2020.00813>
 - [35] Dave S. Kerby. 2014. The Simple Difference Formula: An Approach to Teaching Nonparametric Correlation. *Comprehensive Psychology* 3 (2014). <https://api.semanticscholar.org/CorpusID:120622013>
 - [36] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding deep learning system testing using surprise adequacy. In *Proceedings of the 41st International Conference on Software Engineering (Montreal, Quebec, Canada) (ICSE '19)*. IEEE Press, 1039–1049. <https://doi.org/10.1109/ICSE.2019.00108>
 - [37] Matt Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (Long Beach, California, USA) (NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 4069–4079.
 - [38] Stefano Carlo Lambertenghi, Hannes Leonhard, and Andrea Stocco. 2025. Benchmarking Image Perturbations for Testing Automated Driving Assistance Systems. In *Proceedings of the 18th IEEE International Conference on Software Testing, Verification and Validation (ICST '25)*. IEEE.
 - [39] S. Lapuschkin, S. W  ldchen, A. Binder, G. Montavon, W. Samek, and K.-R. M  ller. 2019. Unmasking Clever Hans predictors and assessing what machines really learn. <https://doi.org/10.1038/s41467-019-08987-4>
 - [40] RoboFlow User Idrago. 2022. Dogs and Cats Classifier Dataset. Kaggle. <https://www.kaggle.com/datasets/rajarshi2712/dogs-and-cats-classifier>
 - [41] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll  r, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision (ECCV)*. Springer, 740–755. https://doi.org/10.1007/978-3-319-10602-1_48
 - [42] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.
 - [43] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: multi-granularity testing criteria for deep learning systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (Montpellier, France) (ASE '18)*. Association for Computing Machinery, New York, NY, USA, 120–131. <https://doi.org/10.1145/3238147.3238202>
 - [44] Maryam, Matteo Biagiola, Andrea Stocco, and Vincenzo Riccio. 2025. Benchmarking Generative AI Models for Deep Learning Test Input Generation. In *Proceedings of 18th IEEE International Conference on Software Testing, Verification and Validation (ICST '25)*. IEEE, 12 pages.

- [45] Maryam Maryam, Matteo Biagiola, Andrea Stocco, and Vincenzo Riccio. 2026. GIFTbench: Generative image fuzz testing benchmark. *Science of Computer Programming* (2026).
- [46] Raghav Mehta, Vitor Albiero, Li Chen, Ivan Evtimov, Tamar Glaser, Zhiheng Li, and Tal Hassner. 2022. You Only Need a Good Embeddings Extractor to Fix Spurious Correlations. <https://doi.org/10.48550/arXiv.2212.06254>
- [47] Giorgi Merabishvili, Oliver Weissl, and Andrea Stocco. 2026. Latent Regularization in Generative Test Input Generation. In *Proceedings of the 48th International Conference on Software Engineering Workshops*.
- [48] K. W. Morton and D. F. Mayers. 2005. *Numerical Solution of Partial Differential Equations: An Introduction*. Cambridge University Press, USA.
- [49] Nusrat Jahan Mozumder, Felipe Toledo, Swaroopa Dola, and Matthew B. Dwyer. 2025. RBT4DNN: Requirements-based Testing of Neural Networks. arXiv:2504.02737 [cs.SE] <https://arxiv.org/abs/2504.02737>
- [50] Neelofar Neelofar and Aldeida Aleti. 2024. Identifying and Explaining Safety-critical Scenarios for Autonomous Vehicles via Key Features. *ACM Transactions on Software Engineering and Methodology* 33, 4 (2024), 1–32.
- [51] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. 2001. On spectral clustering: analysis and an algorithm. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic* (Vancouver, British Columbia, Canada) (NIPS'01). MIT Press, Cambridge, MA, USA, 849–856.
- [52] Augustus Odena, Catherine Olsson, David Andersen, and Ian Goodfellow. 2019. TensorFuzz: Debugging Neural Networks with Coverage-Guided Fuzzing. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 4901–4911. <https://proceedings.mlr.press/v97/odena19a.html>
- [53] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2019. DeepXplore: automated whitebox testing of deep learning systems. *Commun. ACM* 62, 11 (Oct. 2019), 137–145. <https://doi.org/10.1145/3361566>
- [54] Vincenzo Riccio, Gunel Jahangirova, Andrea Stocco, Nargiz Humbatova, Michael Weiss, and Paolo Tonella. 2020. Testing machine learning based systems: a systematic mapping. *Empirical Software Engineering* 25 (2020), 5193–5254.
- [55] Vincenzo Riccio and Paolo Tonella. 2020. Model-based exploration of the frontier of behaviours for deep learning system testing. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 876–888.
- [56] Vincenzo Riccio and Paolo Tonella. 2023. When and Why Test Generators for Deep Learning Produce Invalid Inputs: an Empirical Study. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. 1161–1173. <https://doi.org/10.1109/ICSE48619.2023.00104>
- [57] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 618–626. <https://doi.org/10.1109/ICCV.2017.74>
- [58] Yujun Shen, Ceyuan Yang, Xiaou Tang, and Bolei Zhou. 2020. InterFaceGAN: Interpreting the Disentangled Face Representation Learned by GANs. *TPAMI* (2020).
- [59] Cheng Shi, Yizhou Yu, and Sibe Yang. 2026. Vision Transformers Need More Than Registers. *arXiv preprint arXiv:2602.22394* (2026).
- [60] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).
- [61] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825* (2017).
- [62] Alexander Sorokin and David Forsyth. 2008. Utility data annotation with amazon mechanical turk. In *2008 IEEE computer society conference on computer vision and pattern recognition workshops*. IEEE, 1–8.
- [63] Andrea Stocco, Paulo J. Nunes, Marcelo d’Amorim, and Paolo Tonella. 2022. ThirdEye: Attention Maps for Safe Autonomous Driving Systems. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering (ASE '22)*. IEEE/ACM. <https://doi.org/10.1145/3551349.3556968>

- [64] Andrea Stocco, Michael Weiss, Marco Calzana, and Paolo Tonella. 2020. Misbehaviour Prediction for Autonomous Driving Systems. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE '20)*. ACM, 12 pages. <https://doi.org/10.1145/3377811.3380353>
- [65] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. arXiv:1312.6199 [cs.CV] <https://arxiv.org/abs/1312.6199>
- [66] Qwen Team. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] <https://arxiv.org/abs/2505.09388>
- [67] Yongqiang Tian, Shiqing Ma, Ming Wen, Yepang Liu, Shing-Chi Cheung, and Xiangyu Zhang. 2021. To what extent do DNN-based image classification models make unreliable inferences? *Empirical Softw. Engg.* 26, 5 (Sept. 2021), 40 pages. <https://doi.org/10.1007/s10664-021-09985-1>
- [68] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. <http://jmlr.org/papers/v9/vandermaaten08a.html>
- [69] V. Vapnik. 1991. Principles of risk minimization for learning theory. In *Proceedings of the 5th International Conference on Neural Information Processing Systems* (Denver, Colorado) (NIPS'91). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 831–838.
- [70] Andrey Voynov and Artem Babenko. 2020. Unsupervised discovery of interpretable directions in the GAN latent space. In *Proceedings of the 37th International Conference on Machine Learning (ICML '20)*. JMLR.org, Article 907, 11 pages.
- [71] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. 2018. Transferring GANs: Generating Images from Limited Data. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part VI*. Springer-Verlag, Berlin, Heidelberg, 220–236. https://doi.org/10.1007/978-3-030-01231-1_14
- [72] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. 2020. Towards Fairness in Visual Recognition: Effective Strategies for Bias Mitigation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 8916–8925. <https://doi.org/10.1109/CVPR42600.2020.00894>
- [73] Z. Wang, E.P. Simoncelli, and A.C. Bovik. 2003. Multiscale structural similarity for image quality assessment. In *The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Vol. 2. 1398–1402 Vol.2. <https://doi.org/10.1109/ACSSC.2003.1292216>
- [74] Oliver Weißl, Amr Abdellatif, Xingcheng Chen, Giorgi Merabishvili, Vincenzo Riccio, Severin Kacianka, and Andrea Stocco. 2025. Targeted Deep Learning System Boundary Testing. *ACM Transactions on Software Engineering and Methodology* (2025).
- [75] Oliver Weißl, Vincenzo Riccio, Severin Kacianka, and Andrea Stocco. 2026. HyperNet-Adaptation for Diffusion-Based Test Case Generation. arXiv:2601.15041 [cs.LG] <https://arxiv.org/abs/2601.15041>
- [76] Frank Wilcoxon. 1945. Individual Comparisons by Ranking Methods. *Biometrics Bulletin* 1, 6 (Dec. 1945), 80. <https://doi.org/10.2307/3001968>
- [77] Zongze Wu, Dani Lischinski, and Eli Shechtman. 2021. StyleSpace Analysis: Disentangled Controls for StyleGAN Image Generation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 12858–12867. <https://doi.org/10.1109/CVPR46437.2021.01267>
- [78] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. 2015. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365* (2015).
- [79] Oğuz Kaan Yüksel, Enis Simsar, Ezgi Gülperi Er, and Pinar Yanardag. 2021. LatentCLR: A Contrastive Learning Approach for Unsupervised Discovery of Interpretable Directions. arXiv:2104.00820 [cs.LG]
- [80] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* 48, 1 (2020), 1–36. <https://doi.org/10.1109/TSE.2019.2962027>
- [81] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 2018. DeepRoad: GAN-based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (Montpellier, France) (ASE 2018)*. ACM, New York, NY, USA, 132–142. <https://doi.org/10.1145/3238147.3238187>

- [82] Tahereh Zohdinasab, Vincenzo Riccio, Alessio Gambi, and Paolo Tonella. 2021. Deephyperion: exploring the feature space of deep learning-based systems through illumination search. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '21)*. Association for Computing Machinery, 79–90.

A GENERATED EXAMPLES

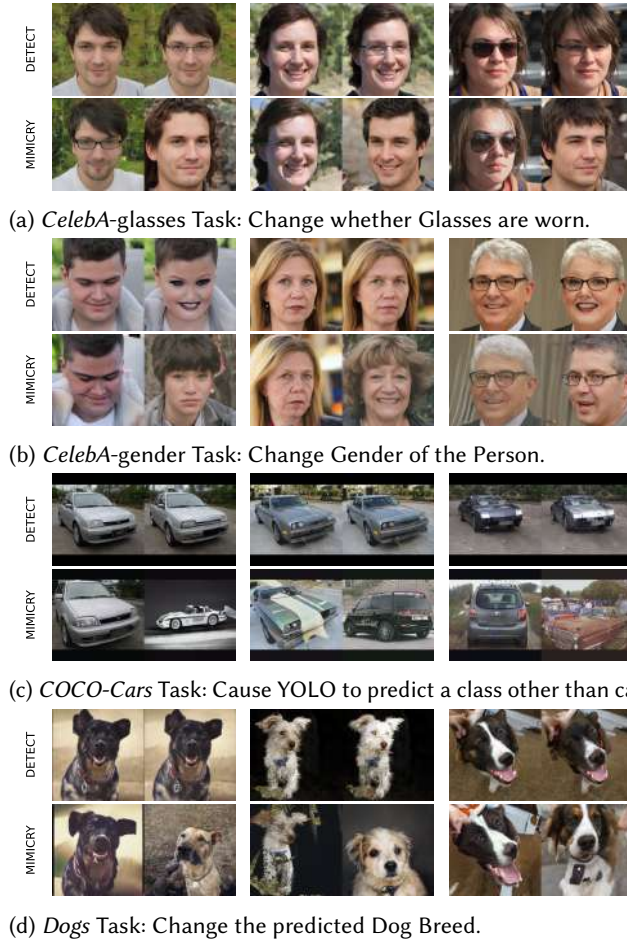


Fig. 11. Original-to-perturbed image pairs generated by DETECT and MIMICRY on each task.

Fig. 11 presents representative example pairs comparing DETECT and MIMICRY. Across tasks, DETECT produces more localized and attribute-specific perturbations, whereas MIMICRY often alters multiple features and introduces additional unintended changes. In general, perturbations generated by DETECT remain more controlled, modifying only the minimal region required for class transition. These examples illustrate the qualitative differences underlying the quantitative trends reported in the main text.

B FULL PROMPT EXAMPLES USED FOR VLM EVALUATION

The following prompts were used for VLM-based relevance attribution and relabeling in the eyeglasses-detection task.

Prompt for relevance attribution.

You are given three AI-generated images:

The first is the original face.

The second is generated by changing one latent direction.

The third is a difference mask showing where the change occurred.

Focus specifically on whether the presence of eyeglasses changes between the first and second images.

Output a JSON object with the following keys:

```
{ "answer": "yes, eyeglasses are added/removed"
  or "no, eyeglasses remain the same." }
```

Use the mask to support your reasoning and ignore irrelevant changes such as lighting or color tone.

Prompt for relabeling.

You are an image tagger for AI-generated portraits.

For each image, output a JSON object with the following fields:

Determine whether the person in the image is wearing eyeglasses.

Output a JSON object.

```
{ "glasses": "Yes / No / Ambiguous" }
```

If the image quality is poor or unclear, respond with "Ambiguous".

C SELECTION OF THE CONFIDENCE DROP THRESHOLD

The selection of the confidence drop threshold for our spurious feature oracle is guided by both the theoretical mechanics and an empirical sensitivity analysis.

Relying on misclassification to define spurious features is inconsistent across varying taxonomy sizes. In binary classification, a change in the predicted class requires the decision logit to cross zero. Conversely, in dense multi-class tasks, misclassification is determined by relative class competition. The target class logit might decrease only slightly or even increase, yet still be overtaken by a competing class. Therefore, we suggest decoupling the measurement of spurious reliance from the volatile mechanics of multi-class boundary crossings and defining the threshold solely on the target class logit.

Fig. 12 provides a descriptive sensitivity analysis. The number of retained unique features decreases as the threshold increases, while the average recurrence of the retained features initially declines and then rises after the most selective region. We use 0.4 as an empirical operating point in the transition region: it removes many

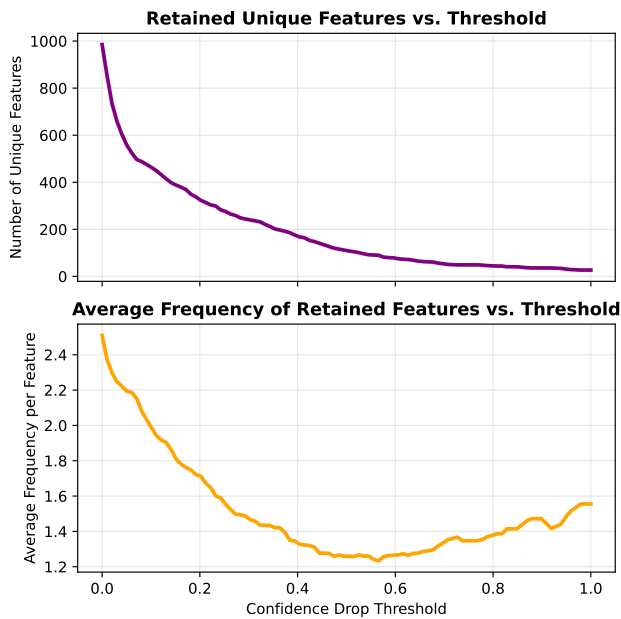


Fig. 12. Sensitivity of feature retention to the confidence-drop threshold. Top: number of retained unique features. Bottom: average frequency per retained feature.

small logit fluctuations while retaining a sufficiently broad set of recurring features for analysis. However, due to the absence of ground-truth spurious-feature labels, this choice is not objectively optimal. Instead, the threshold should be treated as a configurable design choice that may vary across SUTs and tasks according to the desired trade-off between coverage and selectivity.