INTROSPECTIVE SIGNALS FOR FAILURE PREDICTION IN VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing approaches for failure prediction in VLAs rely on information derived from policy inputs, outputs, and latent representations. However, the predictive value of relationships among *internal* signals (such as attention, value vectors, and hidden states) and their evolution during execution remains underexplored. We hypothesize that these patterns can support early and accurate prediction of a VLA's eventual failure. We construct hypothesis-driven introspective signals organized around four categories – *world sensitivity*, *temporal consistency*, *attention–attribution agreement*, and *internal coordination* – and learn temporal patterns over them that align with with episode-level outcome labels. Across three policy families and two LIBERO-based task settings, our models predict failures earlier and more accurately than existing methods and generalizes to unseen tasks.

1 INTRODUCTION

Robot policy failures can be costly and difficult to recover from. The scope of this challenge expands as robot learning moves toward generalist policies designed to operate across diverse tasks and environments. Recent vision-language-action (VLA) models have demonstrated increasingly broad capabilities, from simple pick-and-place tasks to performing extended household tasks in previously unseen environments (Amin et al., 2026; Kim et al., 2025b). Yet neither this breadth of capability nor strong benchmark performance ensures reliable execution (Gu et al., 2025; Fei et al., 2026). While earlier failure prediction methods have often been developed and evaluated for individual tasks (Gokmen et al., 2023; Xu et al., 2025), generalist policies call for predictors that can operate across tasks without task-specific retraining (Gu et al., 2025). Predicting eventual failure during execution could inform decisions about whether to stop, retry, initiate recovery, or request human assistance (Sinha et al., 2024; Farid et al., 2022; Liu et al., 2025; 2024; Karli et al., 2026).

Existing approaches seek evidence of failure in policy observations, generated actions, and latent representations. Observation-based approaches detect departures from the distribution of successful executions (Xu et al., 2025; Römer et al., 2025). Such departures can provide warning signs, but unfamiliar observations don't necessarily mean eventual failure, particularly for policies designed to generalize beyond their training conditions (Römer et al., 2025). Other approaches estimate uncertainty in action generation (Karli et al., 2026; Römer et al., 2025) or learn failure classifiers directly over model hidden states (Gu et al., 2025). However, their generalizability to *unseen* tasks has not been evaluated. In each case, a single source is scored in isolation at each step, but failure may first appear as a breakdown in how internal quantities relate to one another and to the world rather than any one quantity becoming unusual as previous works rely on. The predictive value of explicitly characterizing such relationships among internal model signals, and their evolution during execution, remains underexplored.

In this paper, we pose an extensive set of features that reflect how attention is distributed across input modalities, layers, and heads within a VLA's transformer model, and how these distributions change across execution steps. We then train gated recurrent unit (GRU) models based on each feature to evaluate their ability to predict the robot's eventual success/failure at the task. By evaluating these features across π_0-FAST Pertsch et al. (2025), $\pi_{0.5}$ Black et al. (2025), and OpenVLA-OFT Kim et al. (2025a), we identify features that offer robust success/failure prediction across multiple VLA models and to unseen task variations.

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

In summary, our contributions are:

- We formulate four hypotheses about differences in the internal processing patterns of successful and failing VLA executions, and construct introspective signals to characterize these differences.
- We model temporal patterns from individual and grouped introspective signals using only episode-level outcome labels to predict eventual failure from partial rollouts.
- We conduct comprehensive evaluations across *three* VLA policy families, *two* LIBERO-based task settings, and both *seen* and *unseen* task evaluation protocols, assessing *per-step* and *cumulative* prediction performance using complementary *threshold-independent* and *threshold-based* metrics.
- We analyze which introspective signals carry complementary failure evidence within each category, how these signals differ across VLA architectures, and how prediction performance scales with the number of signals, providing insights into internal processing patterns associated with eventual failure.

2 Related Work

Failure prediction for robot policies. Existing approaches predict robot policy failures through out-of-distribution (OOD) detection, uncertainty estimation, or classifiers trained directly on model latent representations. OOD detection methods flag observations or embeddings that depart from the distribution seen during training or successful execution (Farid et al., 2023; 2022).

FAIL-Detect (Xu et al., 2025) follows this approach, learning a density-based score from successful rollouts alone, and model-based monitors extend it by imagining future states in a learned world model and flagging those that appear risky (Liu et al., 2024; 2025). FIPER (Römer et al., 2025) raises an alarm when an embedding novelty score and the spread of sampled action chunks are jointly high, and Sentinel (Agia et al., 2025) similarly monitors generative policies without failure data by measuring the temporal consistency of predicted action distributions. INSIGHT (Karli et al., 2026) learns which patterns of token-level predictive uncertainty precede failure in autoregressive VLAs. Classifier-based methods learn failure directly from labeled rollouts, for example by approximating a value function (Gokmen et al., 2023) or, as in SAFE (Gu et al., 2025), by training a multitask failure detector on a VLA's hidden-state features.

A complementary line of work uses foundation models to recognize and explain failures once they occur (Liu et al., 2023b; Duan et al., 2024; Sinha et al., 2024). These approaches draw on observations, action outputs, and latent representations as largely separate sources of evidence, each summarized by a single score or feature vector at each step. However, the predictive value of relationships among internal signals and their evolution during execution remains underexplored.

Internal signals in large models. We explore failure-predictive signals inspired by interpretability research on transformers. Attention analyses show that heads specialize in linguistic and cross-modal relations (Clark et al., 2019; Li et al., 2020; Voita et al., 2019), and norm-based analyses show that value vectors must be considered alongside attention weights (Kobayashi et al., 2020). Gradient-based attribution methods estimate input influence directly (Simonyan et al., 2014; Selvaraju et al., 2017; Ancona et al., 2018), and attention weights often disagree with such attribution estimates (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019).

Work on language and vision-language models further shows that internal states anticipate errors. Probes on hidden states predict whether a statement is true (Azaria & Mitchell, 2023), the geometry of internal embeddings indicates hallucination risk (Chen et al., 2024), the ratio of attention paid to context versus generated tokens detects contextual hallucinations (Chuang et al., 2024), and abnormal attention aggregation precedes hallucination in multimodal models (Huang et al., 2024). These methods target single discrete predictions, whereas VLA failures develop gradually through closed-loop execution, so whether internal signals predict failures remains largely unexplored.

3 Problem Formulation

We consider a VLA policy executing a task over an episode of T policy inference steps. The episode length T may vary depending on whether execution terminates early or reaches the maximum allowed number of steps. At each step $t \in \{1, \ldots, T\}$, the policy receives an observation

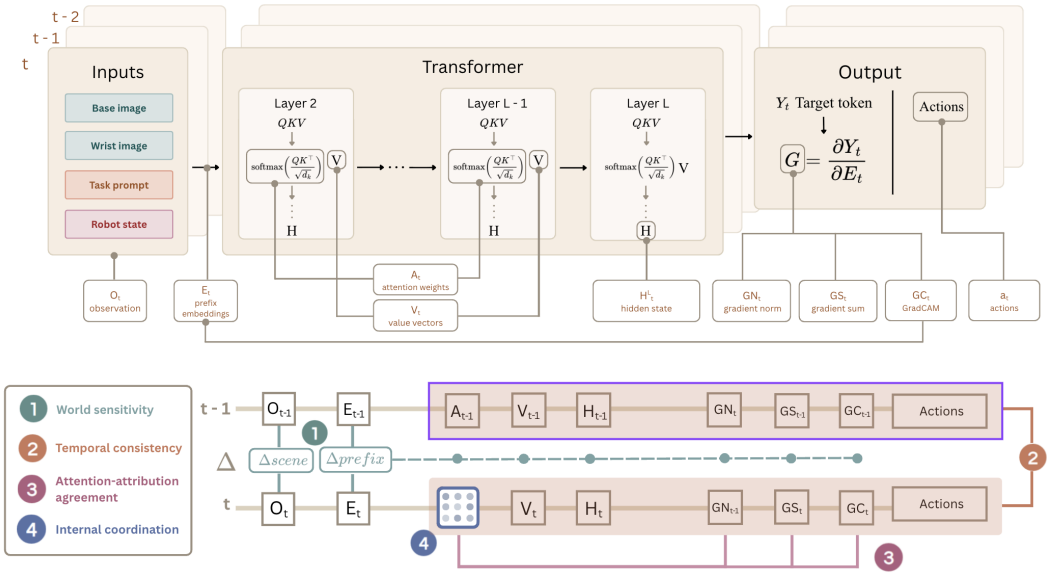

Figure 1: Signals collected during VLA inference and composition of feature categories.

$o_t = (l, I_t, q_t)$, where l is the language instruction, I_t denotes the RGB camera images, and q_t is the robot state, and predicts an action chunk a_t. A rollout episode is represented as $\tau = \{(o_t, a_t)\}_{t=1}^{T}$ and has a binary outcome label $y \in \{0, 1\}$, where $y = 1$ indicates failure and $y = 0$ success.

Our goal is to continuously predict the eventual episode outcome at each timestep t using the information from steps $1...t$. In addition to the rollout prefix $\tau_{\leq t}$, the predictor has access to internal quantities $r_{\leq t} = \{r_s\}_{s=1}^{t}$ recorded during policy inference, such as attention weights, attribution estimates, and latent representations. We formulate failure prediction as binary classification over these prefixes: $\hat{y}_t = f(\langle \tau, r \rangle_{\leq t}) \in \{0, 1\}$, where $\hat{y}_t = 1$ predicts eventual failure and $\hat{y}_t = 0$ predicts eventual success. Only episode-level outcome labels are available, so each prefix is supervised by the final outcome y. Our objective is to predict eventual failure as early and accurately as possible while maintaining predictive performance on tasks unseen during predictor training.

4 Approach

We use Gated Recurrent Unit (GRU) models to learn relationships among signals collected during policy inference and how changes in these relationships correlate with eventual success or failure of the rollout. We distinguish *signals* (the quantities we collect or derive from the VLA during action prediction) from *features* (the relationships we construct from these signals).

Our choice of signals is informed by prior work on how neural networks process inputs and how those inputs influence their predictions. Studies of attention show how models attend to language tokens and image regions (Clark et al., 2019; Li et al., 2020). We collect *attention weights* to examine how attention is distributed across inputs and how it changes during execution. We also collect *value vectors*, which contain the information combined using these weights (Kobayashi et al., 2020). Following work on *gradient-based attribution*, we compute gradients with respect to input embeddings and combine these gradients with the embeddings to measure each input token's contribution to a selected prediction score (Simonyan et al., 2014; Ancona et al., 2018). We then examine whether comparisons across signals, layers, heads, and inference steps help predict eventual task failure.

4.1 Signals

Collected signals. As shown in Figure 1, at each inference step t, we record the **observation** o_t, **prefix embeddings** E_t, second- and penultimate-layer **attention weights** A_t^ℓ and **value vectors** V_t^ℓ,

final-layer **hidden states** H_t, and **predicted actions** a_t. Prefix embeddings E_t are the token representations of the input observation o_t supplied to the transformer backbone before action prediction. They include images and task language instruction embeddings, together with state embeddings when the policy includes state in its prefix. Attention weights and value vectors correspond to A and V in the attention computation (Vaswani et al., 2017):

$$A = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right), \qquad \text{Attention}(Q, K, V) = AV. \tag{1}$$

We denote the recorded attention weights and value vectors by $A_t^{\ell,h}$ and $V_t^{\ell,h}$, where ℓ and h index the layer and attention head, respectively.

Derived attribution signals. During inference, we select a scalar quantity Y_t from action predictions: the first selected action token's logit for π_0-FAST, the sum of predicted action velocities at one flow step for $\pi_{0.5}$, and the sum of predicted actions for OpenVLA-OFT. The **embedding gradient** G_t is the gradient of Y_t with respect to the prefix embeddings E_t:

$$G_t(j, c) = \frac{\partial Y_t}{\partial E_t(j, c)}, \tag{2}$$

where j indexes a token and c an embedding dimension. The gradient therefore follows the computation through the policy from the prefix embeddings to the selected output score, measuring how sensitive Y_t is to changes in each of the embedding components.

We summarize the embedding gradient for each token using its norm and sum over embedding dimensions, as **embedding gradient norm** GN_t and **embedding gradient sum** GS_t:

$$\text{GN}_t(j) = \sqrt{\sum_c G_t(j, c)^2}, \qquad \text{GS}_t(j) = \sum_c G_t(j, c). \tag{3}$$

We use $|\text{GS}_t(j)|$ when comparing token attribution values. We also combine each token's gradient with its embedding and retain the positive part (Selvaraju et al., 2017):

$$\text{GC}_t(j) = \max\left(0, \sum_c G_t(j, c) E_t(j, c)\right). \tag{4}$$

Each of GN, GS, and GC assigns an attribution value to each image, task, or state token.

4.2 Feature Construction

Prior work examines relationships between neural network signals and inputs, including comparisons of attention weights with gradient-based attribution (Jain & Wallace, 2019) and how input changes affect feature attribution scores Ghorbani et al. (2018). Other studies explore structural relationships within networks, such as similarities among attention heads Clark et al. (2019) and how different layers capture distinct information Tenney et al. (2019); these works motivate our cross-signal, multi-head, multi-layer analysis of VLA models to identify patterns associated with task failure.

We construct features by comparing the value of two signals (a and b) at two timesteps (i and j); i.e., $x_t = \text{Distance}(s_i^a, s_j^b)$. For comparisons at the same inference step, s_t^a and s_t^b may be the same signal type from different layers where, the same signal type from different attention heads within a layer, or different signal types. For comparisons between consecutive steps, the two arguments are the same signal type evaluated at $i = t$ and $j = t - 1$, respectively. We consider multiple distance measures: cosine similarity, top-k *intersection-over-union (IoU)*, Spearman correlation, and Wasserstein distance.

Hypotheses. We select signals and timesteps to construct features according to four hypotheses about differences between successful and failing executions. (1) *World sensitivity* hypothesizes that they differ in how changes in attention, attribution, and latent representations relate to changes in visual observations and robot state (Table 1); (2) *Temporal consistency* hypothesizes that they differ in the consistency of attention and attribution across inference steps and in the agreement between

Table 1: World sensitivity features.

Feature	Description	Formula	Variants s	Variants m
`obs_change_{m}`	How much the input changed	$\text{mean}\,\lvert z_{t,m} - z_{t-1,m}\rvert$	—	$\mathcal{O}$
`embedding_cos_{m}`	How similar input embeddings are across steps	$\cos(p_{t,m}, p_{t-1,m})$	—	$\mathcal{O}$
`signal_cos_{s,m}`	How similar a signal is across steps	$\cos(x^s_{t,m}, x^s_{t-1,m})$	$\mathcal{S}$	$\mathcal{M}$
`changed_region_cos_{s,m}`	Signal similarity where the image changed most	$\cos(x^s_t[R_t], x^s_{t-1}[R_t])$	$\mathcal{S}$	$\{\text{base}\}$

Table 2: Temporal consistency features.

Feature	Description	Formula	Variants s	Variants m	Variants d
`replan_raw`	How much the action plan changed	$r_t = \text{RMS}(a_{t-1}[e:] - a_t[:H-e])$	—	—	—
`relative_replan`	Plan change relative to image change	$\log \frac{r_t}{\Delta o_t + \epsilon}$	—	—	—
`temporal_cos_{s,m}`	How similar attention is across steps	$\cos(x^s_{t,m}, x^s_{t-1,m})$	$\{\text{aw}\}$	$\mathcal{M}$	$\{\text{cos}\}$
`temporal_{d,s,m}`	How similar maps are across steps	$d(x^s_{t,m}, x^s_{t-1,m})$	$\mathcal{A}$	$\mathcal{M}$	$\mathcal{D}$

consecutive action predictions over overlapping future steps (Table 2); (3) *Attention–attribution agreement* hypothesizes that they differ in the correspondence between attention weights and attribution values, and in agreement among attribution measures (Table 3); (4) *Internal coordination* hypothesizes that they differ in how attention and attribution are distributed across modalities, how value representations align across modalities and layers, and how attention patterns agree across heads and layers (Table 4).

Notation. Feature names use indices for signal type s, modality m, and comparison measure d. The variant sets are:

$$\mathcal{M} = \{\text{base}, \text{wrist}, \text{task}, \text{state}\}, \qquad \mathcal{O} = \{\text{base}, \text{wrist}, \text{state}\},$$
$$\mathcal{C} = \{\text{base}, \text{wrist}\}, \qquad \mathcal{A} = \{\text{A}, \text{GC}, \text{GN}, \text{GS}\},$$
$$\mathcal{S} = \mathcal{A} \cup \{\text{v}, \text{h}\}, \qquad \mathcal{P} = \{\{u, v\} : u, v \in \mathcal{A},\ u \neq v\},$$
$$\mathcal{D} = \{\text{IoU}, \text{Spearman}, \text{EMD}\}, \qquad \mathcal{R} = \{\cos, \text{Spearman}\}.$$

EMD denotes Wasserstein distance. $x^s_{t,m}$ denotes signal s for modality m at step t. $z_{t,m}$ is the channel-averaged image or proprioceptive state, $p_{t,m}$ the modality's prefix embeddings, and R_t the ten base-camera patches with the largest image change. H is the action-chunk length, e the number of executed actions, and Δo_t the base-camera image change. $b^s_{t,m}$ is modality m's share of the total attention or attribution for signal s at step t, $c^s_{t,m}$ is its centered log-ratio, and $\mathbf{c}^s_t$ collects these log-ratios across modalities. ℓ_s, ℓ_d denote the second and penultimate layers; $u_{h,m}$ is head h's query-averaged attention distribution over modality m, and H_{norm} denotes normalized entropy. Supported combinations depend on the VLA model; a dash indicates no additional variant.

4.3 Model Architecture and Training

For each policy family, we train a separate predictor for each signal category to evaluate its predictive contribution. Let $x_{i,t}$ collect the selected signals within that category for episode i at step t, and let

Table 3: Attention–attribution agreement features.

Feature	Description	Formula	Variants		
			Pair	m	d
`agreement_cos_`$\{u, v, m\}$	How closely two signals agree	$\cos(x^u_{t,m}, x^v_{t,m})$	$\{u, v\} \in \mathcal{P}$	$\mathcal{M}$	$\{\cos\}$
`agreement_`$\{d, u, v, m\}$	How closely two maps agree	$d(x^u_{t,m}, x^v_{t,m})$	$\{u, v\} \in \mathcal{P}$	$\mathcal{M}$	$\mathcal{D}$

Table 4: Internal coordination features.

Feature	Description	Formula	Variants		
			s	m	d / ℓ
`relative_budget_`$\{s, m\}$	How much a modality receives relative to others	$c^s_m = \log b^s_m - \text{mean}_{m'} \log b^s_{m'}$	$\mathcal{A}$	$\mathcal{M}$	—
`budget_change_`$\{s\}$	How much the split across modalities changed	$\|\mathbf{c}^s_t - \mathbf{c}^s_{t-1}\|_2$	$\mathcal{A}$	—	—
`image_text_balance_`$\{s\}$	How much images receive relative to text	$\log \frac{b^s_{\text{base}} + b^s_{\text{wrist}}}{b^s_{\text{task}}}$	$\mathcal{A}$	—	—
`cross_layer_`$\{d, m\}$	How closely two layers agree	$d(x^{\text{A}}_{\ell_s,m}, x^{\text{A}}_{\ell_d,m})$	$\{\text{A}\}$	$\mathcal{M}$	$\mathcal{R}$
`head_agreement_`$\{m\}$	How similarly attention heads distribute attention	$\text{mean}_{h<h'} \cos(u_{h,m}, u_{h',m})$	$\{\text{A}\}$	$\mathcal{M}$	—
`head_concentration_`$\{m, \ell\}$	How much attention falls on a few tokens	$1 - \text{mean}_h H_{\text{norm}}(u_{h,m})$	$\{\text{A}\}$	$\mathcal{C}$	$\{\ell_s, \ell_d\}$

$X_i = \{x_{i,t}\}_{t=1}^{T_i}$ denote their sequence. Each vector uses observations, predicted actions, and internal quantities available through that step. We use at most 60 policy inference steps per episode; most successful rollouts in our LIBERO-based evaluations finish within this horizon.

Recurrent predictor. A single-layer, unidirectional gated recurrent unit (GRU) with 32 hidden units aggregates the preprocessed signal vectors $\widetilde{x}_{i,t}$:

$$h_{i,t} = \text{GRU}_\theta(\text{Dropout}(\widetilde{x}_{i,t}), h_{i,t-1}) \qquad p_{i,t} = \sigma\big(w^\top h_{i,t} + b\big) \tag{5}$$

The score $p_{i,t}$ estimates eventual episode failure using information through step t.

Training objective. Each prefix is supervised by the episode outcome y_i. We minimize class-weighted binary cross-entropy over valid steps in a batch $\mathcal{B}$:

$$\mathcal{L} = -\frac{1}{\sum_{i\in\mathcal{B}} T_i} \sum_{i\in\mathcal{B}} \sum_{t=1}^{T_i} \left[\omega y_i \log p_{i,t} + (1 - y_i) \log(1 - p_{i,t})\right], \tag{6}$$

where $\omega = N_{\text{success}}/N_{\text{failure}}$ is the ratio of successful to failed training episodes.

5 Evaluation

In the following sections, we introduce the baselines (§5.1), describe the experimental setup (§5.2), and present evaluation results (§5.3), examining failure prediction accuracy and timeliness across three VLA policies, two benchmarks, and both familiar and unseen tasks.

5.1 Baselines

We compare our models against four runtime failure predictors that span the range of available supervision and signal type. **FAIL-Detect** (Xu et al., 2025) fits a density model to successful execution

and flags observations that are unlikely under it, using its logpZO score, the most consistent variant in the original work. **FIPER** (Römer et al., 2025) is also failure-data-free but combines two signals, raising an alarm only when an observation-novelty score and an action-uncertainty score fire together. **INSIGHT** (Karli et al., 2026) reads the policy's own token-level predictive uncertainty and learns, from episode outcomes, which uncertainty patterns precede failure. **SAFE** (Gu et al., 2025) is supervised on trajectory-level success and failure labels and operates directly on the VLA's internal representations. Table 5 in Appendix A.1 summarizes the original evaluation setup of each of the methods. Detailed architectures, features, training configurations, and the points at which our implementations depart from the original works are given in Appendix A.

5.2 Experiment Setup

Policies, benchmarks, and baseline coverage. We evaluate our models and the baselines on rollouts from three VLA policies: π_0-FAST (Pertsch et al., 2025), $\pi_{0.5}$ (Black et al., 2025), and OpenVLA-OFT (Kim et al., 2025a), using LIBERO-10 (Liu et al., 2023a) and LIBERO-Plus LIBERO-Plus. We exclude the language variations in LIBERO-Plus because we identified corrupted language instructions in those variations. Our models, FAIL-Detect, and SAFE are evaluated on all three policies. FIPER requires repeated action-chunk samples at a fixed observation, expressed in end-effector coordinates, which are only from $\pi_{0.5}$. INSIGHT requires predictive distributions over discrete action tokens and therefore can only be evaluated on π_0-FAST. Comparisons use the same test rollouts for methods evaluated on a given policy.

Data splits and training protocols. We evaluate both new rollouts of familiar tasks and rollouts of unseen tasks. Under the *episode-random* protocol, complete episodes are randomly partitioned, allowing the same task identities to appear in development and test data. Under the *task-disjoint* protocol, task identities are partitioned, so test tasks are absent from predictor training, model selection, and calibration. Each LIBERO-Plus variation is grouped with its corresponding original task. We use 40 repeated splits for each protocol, with shared test partitions across methods. All prefixes from an episode remain in the same partition. We reserve 20% of episodes for testing under the *episode-random* protocol and *two* of the ten task identities under the *task-disjoint* protocol. Each baseline follows its original training and validation procedure: an 80:20 (%) train+validation:test split, or a 60:20:20 (%) training:validation:test split with a separate validation partition. A pilot comparison of these two procedures for our models produced similar prediction performance, so we use the 80:20 allocation.

Threshold calibration. For comparisons under the original evaluation procedures, we retain the baselines' reported calibration methods: time-varying conformal thresholds for FAIL-Detect, FIPER, and SAFE, and a fixed threshold of 0.5 for INSIGHT. We additionally evaluate all five methods using a common calibration procedure. At each inference step, we select separate thresholds that maximize accuracy, failure-class F1, and balanced accuracy on the designated calibration data. We also select a threshold from successful calibration episodes to target a false-positive rate of 10%. This operating point measures how many failures a method identifies while limiting false alarms on successful episodes.

Evaluation metrics. We evaluate predictions at both the *episode* and *inference-step* levels under both split protocols. Episode-level evaluation follows the baselines' original aggregation procedures, including whether an alarm occurs at any point during an episode; INSIGHT's step-level results are also evaluated under its reported protocol. In our per-step evaluation, each method predicts eventual episode success or failure using information available through the current step. We report *accuracy*, *F1 score*, and *balanced accuracy*. Failure is the positive class throughout. Per-step metrics include only episodes that have an observation at that step. We plot these metrics against policy inference step and report their mean and standard deviation across the 40 splits.

5.3 Experiment Results

We first evaluate a compact predictor built from the six signals with the highest training-set AUROC across our feature categories. This fixed six-signal set is used for the main baseline comparison, while later analyses study individual categories, complementary signal selection, and the effect of feature-set size. Across both *episode-random* and *task-disjoint* evaluation, Our compact

model achieves state-of-the-art failure-prediction performance among the applicable baselines. We compare against SAFE, FAIL-Detect, FIPER, and INSIGHT wherever each method is applicable.

Task-disjoint evaluation reveals a clear generalization gap in prior failure predictors. As shown in Figure 2, while SAFE and FAIL-Detect perform strongly on held-out rollouts from seen tasks, their AUROC drops consistently when the test task is excluded from predictor training, with FAIL-Detect falling to around chance or below for all three VLAs. SAFE degrades more moderately but still loses substantial performance, especially for π_0-FAST. In contrast, INSIGHT and FIPER are more stable in the settings where they apply, and our method maintains the strongest AUROC across most of the rollout under task-disjoint evaluation. These results suggest that failure cues learned from seen tasks do not always transfer reliably to new task identities, making task-disjoint evaluation important for measuring whether a failure predictor captures task-generic rather than task-specific signals. We present results in accuracy, F1 scores, and balanced accuracy in Appendix B.

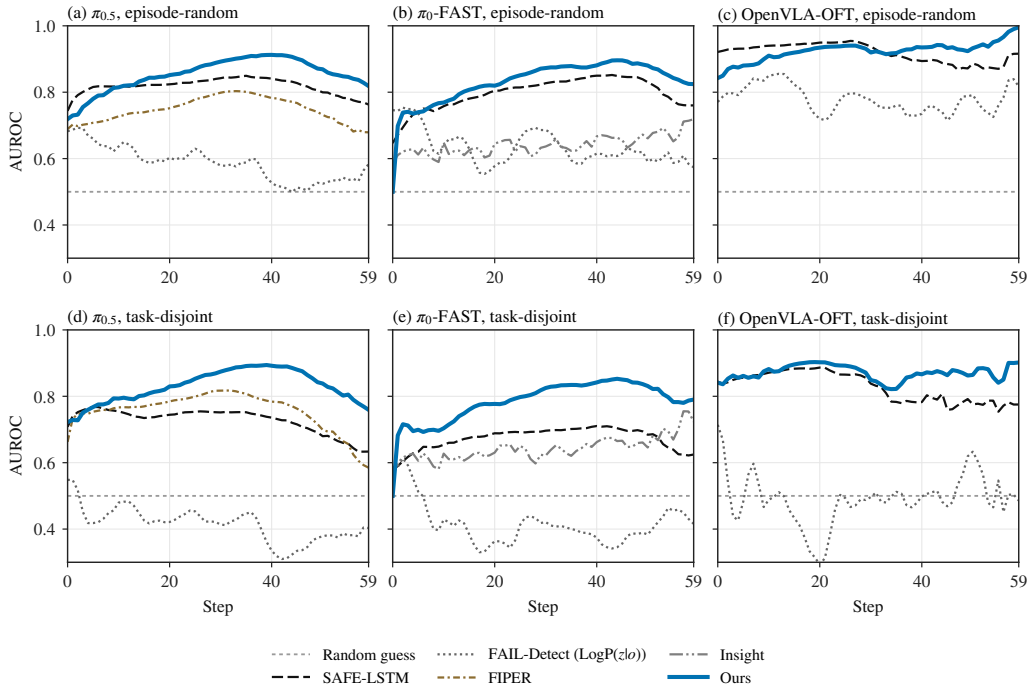

Figure 2: Baseline failure predictors generalize less reliably to unseen tasks. Per-step AUROC is shown for episode-random (top) and task-disjoint (bottom) splits across three VLA families. Most baselines decline when evaluated on unseen tasks, whereas our method retains strong performance across the rollout.

6 Feature Ablation Analysis

To identify which features carry complementary failure evidence, we perform a backward-elimination ablation separately for each VLA model and each feature category. All experiments use the task-disjoint 8:2 setting with 40 folds, and each fold runs its own elimination chain independently and based on only training-set performance. Each chain starts from the 15 features of the category with the highest training-set AUROC. At each step we train a GRU on the current feature set and then measure how much each remaining feature contributes. We replace one feature at a time with an uninformative filler, either its training-set mean or zero, whichever hurts performance less, and re-score the trained model on the training data without retraining. The feature whose removal degrades performance the least is dropped, and we repeat until one feature remains. This gives, for every fold, an ordered elimination path from 15 features to 1, and a GRU trained at every set size (3 models $\times$ 4 categories $\times$ 40 folds $\times$ 15 sizes $=$ 7,200 GRUs). Through this ablation, we

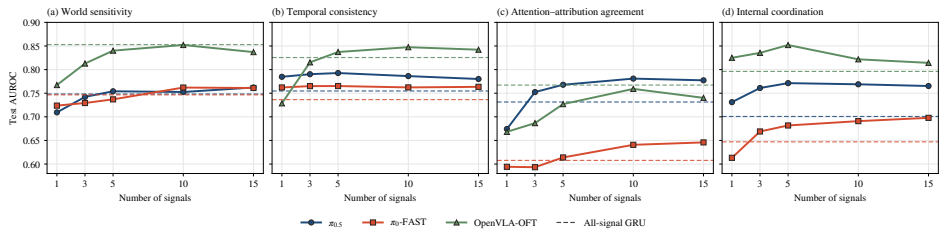

Figure 3: # of features vs failure prediction performance (AUROC) within each feature category

find that (i) each category's surviving features are complementary rather than redundant, except in Attention–attribution agreement, (ii) the specific surviving signals differ across VLA architectures while the high-level relationships are shared, and (iii) failure-prediction performance saturates after a handful of features.

6.1 Results and Discussion

Which features provide complementary failure evidence within each category? Complementarity comes from features that measure different aspects of failure. In world sensitivity, obs_change_m features are paired with features describing how the model's internal state changes. In temporal consistency, the action-side $replan_raw$ features is consistently paired with attention-consistency features. Internal coordination combines distinct properties of the model's internals, including allocation, head organization, and cross-layer consistency. Attention-attribution Agreement is different: its surviving features are mostly variants of the same attention and embedding gradient relationship and therefore provide less complementary information. This suggests that useful combinations arise from different sources of failure evidence rather than multiple measurements of the same relationship. See Appendix C for a list of top 10 feature survivors counts across 40 folds.

How do complementary features differ across VLA architectures? The high-level relationships are shared, but the specific signals that realize them depend on the VLA architecture. The π models rely more heavily on state-related signals, with state accounting for 39% of survivor slots in π_0-FAST and 24% in $\pi_{0.5}$. OpenVLA-OFT instead relies more on camera channels and gradient-based signals. Temporal consistency provides the clearest common structure: $replan_raw$ survives in every fold for all three models, while the complementary attention signal differs by architecture. Only a small set of patterns, including replanning behavior, attention weights and embedding gradient norm agreement, and head structure, are shared across all three models.

How does failure-prediction performance scale with the number of features? As shown in Fig. 3, performance improves quickly with the first few complementary signals and then saturates. Mean task-disjoint test AUROC increases from 0.715 with one signal to 0.746 with three, 0.762 with five, and 0.769 with ten, before slightly decreasing to 0.766 with fifteen. The marginal gain therefore shrinks from $+0.031$ for 1 to 3 signals to $+0.016$ for 3 to 5 and $+0.007$ for 5 to 10. In 11 of 12 model-category chains, performance is already within 0.01 of its peak using at most eight signals. Small backward-eliminated sets also outperform the all-signal GRU in most cases.

7 Conclusion

We show that relationships among VLA internal signals provide useful patterns for failure prediction. Our hypothesis-driven introspective features span across: world sensitivity, temporal consistency, attention–attribution agreement, and internal coordination. Across three VLA models and two LIBERO-based settings, our method predicts failure earlier and more accurately than prior works, achieving state-of-the-art performance. Our analysis further finds most predictive information comes from small sets of complementary features, and that the specific features vary across architectures. These results highlight structured internal signal patterns as a promising basis for reliable VLA failure prediction.

8 AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, including code for recording internal policy signals, computing features, and baseline implementation. We have not used generative AI tools to generate synthetic data sets, develop theoretical models or conceptual frameworks, propose or refine hypotheses, design research methodology or experiments, clean or reformat datasets, or interpret results. Formulating mathematical claims, writing proofs, translation, and qualitative or thematic data analysis are not applicable to this work.

Additionally, we used generative AI tools to draft and edit parts of the paper for readability, identify and summarize relevant literature, and format references. We have reviewed all AI-assisted work. AI-assisted code was reviewed and tested by the authors, and our baseline implementations were checked against results reported in the original works. All AI-suggested references were verified against their original publications, and descriptions of prior work were checked by the authors against the cited papers. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

9 REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we will publicly release the data collection pipeline, signal construction code, and model training code upon acceptance of the paper.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Ziang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 689–723. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/agia25a.html.

Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace B. Connors, James Darpinian, Karan Dhabalia, Jared Di Carlo, Danny Driess, Michael Robert Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Benjamin Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin Leblanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alexander Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi_{0.6}$: a VLA That Learns From Experience. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.087.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sy21R9JAW.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.68. URL https://aclanthology.org/2023.findings-emnlp.68/.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky.

$\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/black25a.html.

Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. Inside: Llms' internal states retain the power of hallucination detection, 2024. URL https://arxiv.org/abs/2402.03744.

Yung-Sung Chuang, Linlu Qiu, Cheng-Yu Hsieh, Ranjay Krishna, Yoon Kim, and James Glass. Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1419–1436. Association for Computational Linguistics, 2024. URL https://aclanthology.org/2024.emnlp-main.84/.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In Tal Linzen, Grzegorz Chrupała, Yonatan Belinkov, and Dieuwke Hupkes (eds.), *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 276–286, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4828. URL https://aclanthology.org/W19-4828/.

Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation, 2024. URL https://arxiv.org/abs/2410.00371.

Alec Farid, David Snyder, Allen Z. Ren, and Anirudha Majumdar. Failure Prediction with Statistical Guarantees for Vision-Based Robot Control. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi: 10.15607/RSS.2022.XVIII.042.

Alec Farid, Sushant Veer, Divyanshu Pachisia, and Anirudha Majumdar. Task-driven detection of distribution shifts with statistical guarantees for robot learning, 2023. URL https://arxiv.org/abs/2106.13703.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. Libero-plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, June 2026.

Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile, 2018. URL https://arxiv.org/abs/1710.10547.

Cem Gokmen, Daniel Ho, and Mohi Khansari. Asking for help: Failure prediction in behavioral cloning through value approximation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5821–5828, 2023. doi: 10.1109/ICRA48891.2023.10161004.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. Safe: Multitask failure detection for vision-language-action models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 40041–40076. Curran Associates, Inc., 2025. doi: 10.52202/085713-1337. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/392d0d05e2f514063e6ce6f8b370834c-Paper-Conference.pdf.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13418–13427, June 2024.

Sarthak Jain and Byron C. Wallace. Attention is not Explanation. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*

(Long and Short Papers), pp. 3543–3556, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1357. URL https://aclanthology.org/N19-1357/.

Ulas Berk Karli, Ziyao Shangguan, and Tesca Fitzgerald. Insight: Inference-time sequence introspection for generating help triggers in vision-language-action models. *International Conference on Robotics and Automation*, 2026.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. Attention is not only a weight: Analyzing transformers with vector norms. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7057–7075, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.574. URL https://aclanthology.org/2020.emnlp-main.574/.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. What does BERT with vision look at? In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5265–5275, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.469. URL https://aclanthology.org/2020.acl-main.469/.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023a. doi: 10.52202/075280-1939. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

Huihan Liu, Shivin Dass, Roberto Martín-Martín, and Yuke Zhu. Model-based runtime monitoring with interactive imitation learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4154–4161, 2024. doi: 10.1109/ICRA57147.2024.10611038.

Huihan Liu, Yu Zhang, Vaarij Betala, Evan Zhang, James Liu, Crystal Ding, and Yuke Zhu. Multi-task interactive robot fleet learning with visual world models. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4286–4313. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/liu25g.html.

Zeyi Liu, Arpit Bahety, and Shuran Song. Reflect: Summarizing robot experiences for failure explanation and correction, 2023b. URL https://arxiv.org/abs/2306.15724.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient Action Tokenization for Vision-Language-Action Models. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.012.

Ralf Römer, Adrian Kobras, Luca Worbis, and Angela Schoellig. Failure prediction at runtime for generative robot policies. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz,

M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 7631–7670. Curran Associates, Inc., 2025. doi: 10.52202/085713-0260. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/0b7cb3b8cc44e652761245537027db44-Paper-Conference.pdf.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations*, 2014.

Rohan Sinha, Amine Elhafsi, Christopher Agia, Matt Foutter, Edward Schmerling, and Marco Pavone. Real-Time Anomaly Detection and Reactive Planning with Large Language Models. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.114.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1452. URL https://aclanthology.org/P19-1452/.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019. URL https://aclanthology.org/P19-1580/.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation, 2019. URL https://arxiv.org/abs/1908.04626.

Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Miller, Robert Lee, Paarth Shah, Rares Andrei Ambrus, Haruki Nishimura, and Masha Itkina. Can We Detect Failures Without Failure Data? Uncertainty-Aware Runtime Failure Detection for Imitation Learning Policies. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.073.

A BASELINE IMPLEMENTATION

All baselines consume signals recorded during the same policy rollouts used to evaluate our proposed models, so the comparison isolates the failure predictor rather than the data. Within a policy family, every method sees identical episode partitions.

A.1 COMPARISON OF BASELINE METHODS

Table 5: Comparison of prior failure-prediction methods. **Failure data**: whether failed rollouts are required for training. **Test scope**: *Seen-task ID/OOD* denotes the same task identity under in- and out-of-distribution conditions; *Unseen tasks* denotes evaluation on task identities not observed during predictor training; *cross-domain transfer* denotes INSIGHT's real-world → LIBERO evaluation using a different policy checkpoint. **Calibration method** and **threshold**: how binary decisions are obtained at test time. **Prediction target**: *episode failure (ever triggered)* means a rollout is counted as failed once any timestep raises an alarm, whereas *step-level* evaluates the current timestep prediction. **Evaluated policies**: the policy families considered in the original study.

Method	Failure data	Test scope	Calibration method	Threshold	Prediction target	Evaluated policies
FAIL-Detect	No	Seen-task ID/OOD	Functional CP	Time-varying	Episode failure (ever triggered)	Diffusion / flow policies
FIPER	No	Seen-task ID/OOD	Functional CP	Time-varying	Episode failure (ever triggered)	Diffusion / flow policies
INSIGHT	Yes	Seen-task OOD + cross-domain transfer	–	0.5	Step-level	π_0-FAST
SAFE	Yes	Seen / unseen tasks	Functional CP	Time-varying	Episode failure (ever triggered)	OpenVLA, π_0, π_0-FAST

A.2 WHICH BASELINES APPLY TO WHICH POLICIES

Two baselines require a signal that only one policy family exposes, so they cannot be evaluated universally. Table 6 summarizes coverage and the feature each method uses.

Table 6: Policy coverage and feature source for each baseline. FAIL-Detect and SAFE apply to all three policies. FIPER requires sampled Cartesian action chunks and INSIGHT requires an autoregressive token distribution, each of which only one policy family provides.

	π_0-FAST	$\pi_{0.5}$	OpenVLA-OFT
FAIL-Detect (logpZO)	pooled prefix image tokens + state	pooled prefix image tokens + state	pooled prefix image tokens + raw state
SAFE	last aligned prefix token	action-expert suffix hidden state	first action-head input position
FIPER	–	RND embedding + XYZ action-chunk entropy	–
INSIGHT	per-token entropy, NLL, Dirichlet aleatoric/epistemic	–	–

FIPER scores uncertainty over a batch of sampled Cartesian action chunks. This requires a policy that can be sampled repeatedly at a fixed observation and whose action chunks are expressed in end-effector space; of our three policies only $\pi_{0.5}$ records both.

INSIGHT scores the token-level predictive distribution of an autoregressive action decoder. π_0-FAST exposes discrete action tokens autoregressively and therefore exposes the required per-token

distribution; $\pi_{0.5}$ and OpenVLA-OFT use continuous action heads, for which the entropy, log-probability and Dirichlet quantities INSIGHT relies on are undefined.

A.3 FAIL-DETECT (LOGPZO)

Method. FAIL-Detect (Xu et al., 2025) frames failure detection as out-of-distribution detection: failures are unlikely under the distribution of successful execution, so a density model fit to successes alone suffices, with no failure data. We use its logpZO score, the best-performing variant in the original work. A flow-matching network f_θ is trained to transport an observation O to standard Gaussian noise, and a single forward Euler step from $t = 0$ estimates the latent noise O maps to:

$$O_s = O + s\,(Z - O), \quad Z \sim \mathcal{N}(0, I),\ s \sim \mathcal{U}[0, 1], \qquad \mathcal{L}(\theta) = \mathbb{E}\big\| f_\theta(O_s, s) - (Z - O)\big\|_2^2, \tag{7}$$

$$Z_O = O + f_\theta(O, 0), \qquad \text{logpZO}(O) = \|Z_O\|_2^2. \tag{8}$$

A large $\|Z_O\|_2^2$ means the observation does not map into the high-density region of the noise prior and is therefore unlikely under the learned distribution of success.

Observation representation. The original method scores the policy's own conditioning vector. We mean-pool the policy's final prefix hidden states over each active camera's image-token span and concatenates the proprioceptive state, stacked over the current and previous policy call. This yields 4112 dimensions for π_0-FAST and $\pi_{0.5}$ and 8208 for OpenVLA-OFT. For OpenVLA-OFT we concatenate the unnormalized recorded state, since its recorder exposes both raw and normalized state and the raw field is the one this mode is defined against.

Training. The score network is a time-conditioned residual MLP (width 512, depth 4, sinusoidal time embedding of dimension 128; 6.6M parameters), trained with Adam at a constant learning rate 10^{-4}, batch size 128, for 500 epochs, matching the original simulation configuration. It is fit on the successful episodes of the training split only. Input features are standardized per dimension using statistics computed on the training split alone.

A.4 SAFE

Method. SAFE (Gu et al., 2025) is a supervised multitask failure detector operating on a VLA's internal representations. At each policy step it reduces the policy's hidden state to a fixed-dimensional vector and maps the resulting sequence to a scalar failure score with a lightweight recurrent head. Training uses trajectory-level success and failure labels across many tasks, so no annotation of the failure timestep is required. We evaluate the LSTM variant as our primary baseline.

Architecture and training. A single-layer LSTM with hidden dimension 256 and a linear failure head, run over the full step history, producing one failure probability per policy call. Training minimizes class-balanced binary cross-entropy at every timestep, with failure as the positive class, using Adam for 1000 epochs at batch size 512 to match the original implementation.

Policy-specific features. SAFE's feature is the internal representation immediately preceding the policy's first action prediction, which is architecture-specific. Our capture points for each policy are listed in Table 7.

Table 7: Policy-specific internal representations used as inputs to SAFE.

Policy	SAFE feature	Dim.
π_0-FAST	Final aligned VLM-prefix hidden state preceding the first action token	2048
$\pi_{0.5}$	Action expert's final hidden state, pooled over the denoising steps and the action horizon	1024
OpenVLA-OFT	Final prefix hidden state at the position immediately preceding the first action-head prediction, validated against the recorded token spans at every call	4096

A.5 FIPER

Method. FIPER (Römer et al., 2025) predicts failure from two independent signals and raises an alarm only when both fire. The first is observation novelty, measured by random network distillation (RND) in the policy's observation-embedding space: a predictor network is trained to match a fixed random target on successful rollouts, and its residual error grows on observations unlike those seen in training. The second is action uncertainty, measured by the entropy of a batch of action chunks sampled at the current observation. Both scores are aggregated over short causal sliding windows and compared against time-varying thresholds, and the two warnings are combined with a logical conjunction.

Observation channel. Target and predictor multilayer perceptrons with hidden widths $(1024, 2048, 4096)$ and a 128-dimensional output; the predictor adds two further layers of widths $(2048, 1024)$. The score is the unsquared Euclidean distance between predictor and target embeddings. Training uses AdamW (learning rate 10^{-4}, weight decay 10^{-5}, $\varepsilon = 10^{-8}$), cosine learning-rate decay, batch size 256, for 250 epochs, with a 10% inner holdout of the training episodes used for monitoring.

Action channel. At each policy call we sample a batch of action chunks and retain their Cartesian end-effector positions. Treating each step of the prediction horizon separately, each dimension d is partitioned into $N_d = \lceil (\max_j a_{j,d} - \min_j a_{j,d})/(\alpha R_d) \rceil$ bins, where R_d is the range of that dimension over the calibration set and $\alpha = 0.03$ is the cell-size factor. A histogram over the resulting product cells gives occupancy probabilities p_c, and the action-chunk entropy score is the sum over the horizon of $-\sum_c p_c \log_2 p_c$.

Aggregation and alarm. Both channels are summed over causal sliding windows of 25 policy calls (observation) and 50 (action). Each channel receives its own conformal threshold curve, calibrated independently on successful episodes as described in Section A.7, and a failure is predicted only when both channels exceed their thresholds at the same policy call.

A.6 INSIGHT

Method. INSIGHT (Karli et al., 2026) predicts when a policy should request help from token-level uncertainty signals available at inference time. For each generated action token it extracts four quantities: the entropy of the predictive distribution, the negative log-probability of the selected token, and Dirichlet-based estimates of aleatoric and epistemic uncertainty. A compact transformer models the sequence of per-token features within a policy call and emits a step-level probability that help is needed.

Architecture and training. The four-dimensional per-token features are projected to a hidden dimension of 64 and encoded by a two-layer transformer (4 attention heads, feedforward dimension 2048, dropout 0.1) with sinusoidal positional encoding and masked attention pooling over valid tokens, followed by a small MLP head producing one logit per policy call. We use INSIGHT's episode-level (weak) supervision setting, in which step logits are pooled into an episode logit with a temperature-scaled log-sum-exp,

$$\ell_{\text{episode}} \;=\; \tfrac{1}{\lambda} \log \sum_t \exp\bigl(\lambda\, \ell_t\bigr), \qquad \lambda = 6, \tag{9}$$

a smooth approximation to the maximum that implements the multiple-instance assumption that a failed episode contains at least one step needing help. The model is trained end to end with binary cross-entropy against episode outcomes using Adam at learning rate 10^{-4} for 30 epochs.

Thresholding. INSIGHT's original evaluation reports its transformer as a plain binary classifier and specifies no threshold-selection procedure. We therefore report it at a fixed threshold of 0.5 on the step probability, flagging an episode if any step reaches it. For cross-method comparison at a shared operating point we also report INSIGHT under the conformal threshold of Section A.7; we note that in the original work conformal prediction appears as a competing baseline rather than as a calibration of the transformer, so those numbers are our addition and not a reproduction.

A.7 THRESHOLD CALIBRATION

FAIL-Detect, FIPER and SAFE all calibrate time-varying thresholds with the conformal prediction framework. Given per-timestep scores from successful calibration episodes, the set is split into two groups: the first gives a pointwise mean trajectory μ_t and a modulation function σ_t, taken as the pointwise maximum absolute residual after trimming the most extreme episodes; the second gives a scalar band width $h = \mathrm{Quantile}_{1-\alpha}\big(\max_t(s_t - \mu_t)/\sigma_t\big)$. The threshold is $\eta_t = \mu_t + h\,\sigma_t$, and an episode is flagged if its score crosses η_t at any policy call. Scores are oriented so that larger values indicate failure, and calibration uses successful episodes only.

For the cumulative scoring protocol the band is recalibrated on cumulative calibration scores. INSIGHT is additionally reported at its fixed 0.5 threshold as described above.

Under the task-disjoint protocol the calibration and test episodes come from disjoint task identities, so the exchangeability assumption underlying the conformal guarantee does not hold. We report the achieved false-positive rate alongside every threshold-based result so this gap is visible.

B BASELINES COMPARISON

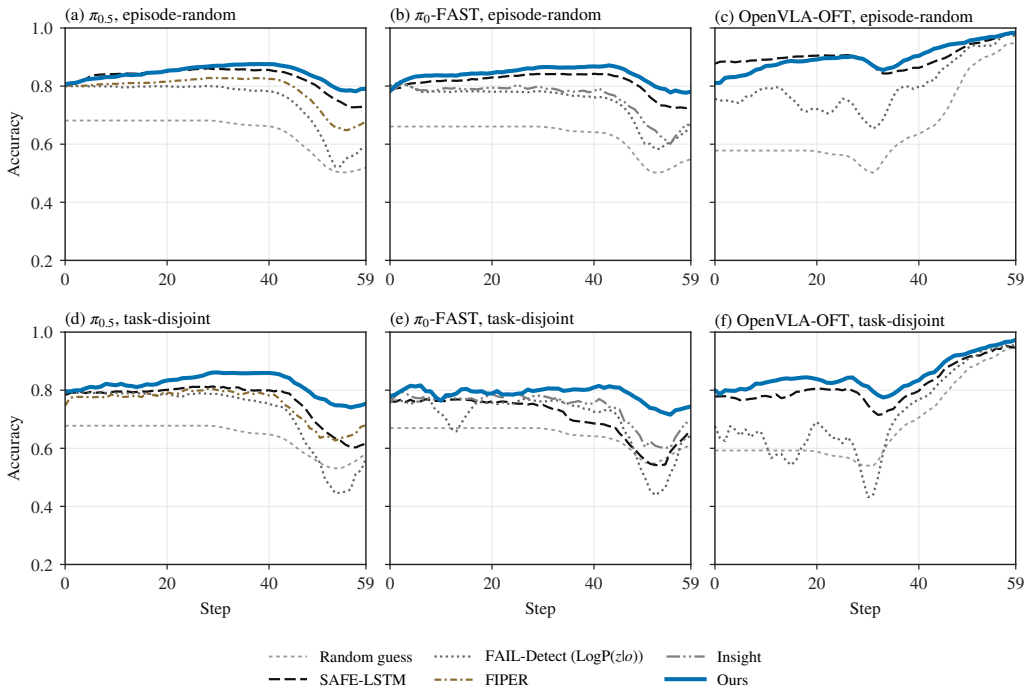

Figure 4: Per-step accuracy is shown for episode-random (top) and task-disjoint (bottom) splits across three VLA families. Most baselines decline when evaluated on unseen tasks, whereas our method retains strong performance across the rollout.

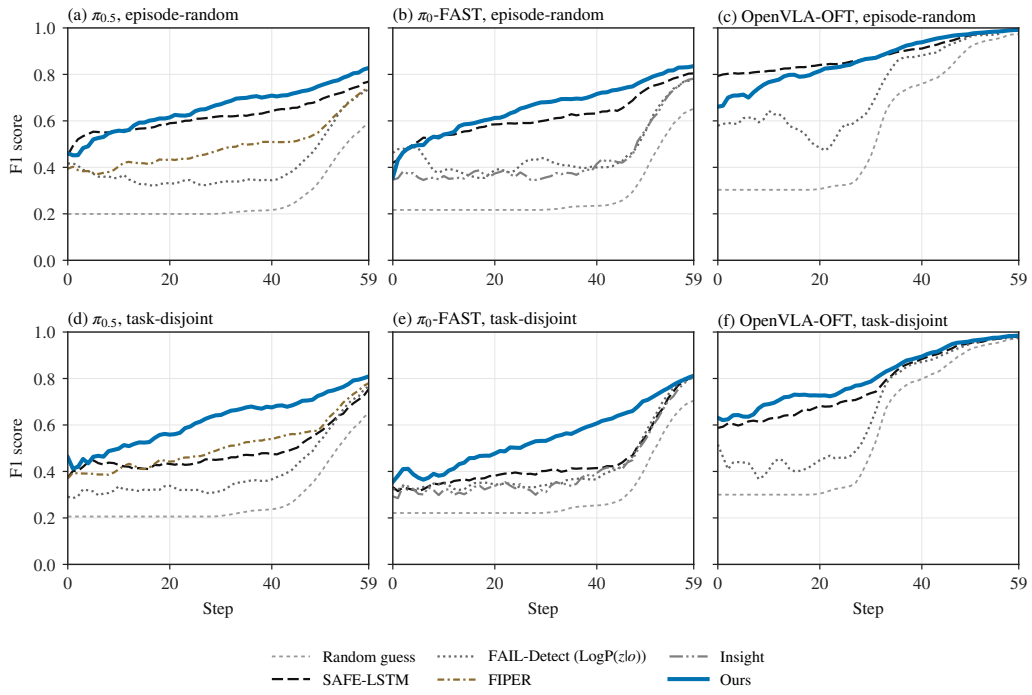

Figure 5: Per-step F1 is shown for episode-random (top) and task-disjoint (bottom) splits across three VLA families. Most baselines decline when evaluated on unseen tasks, whereas our method retains strong performance across the rollout.

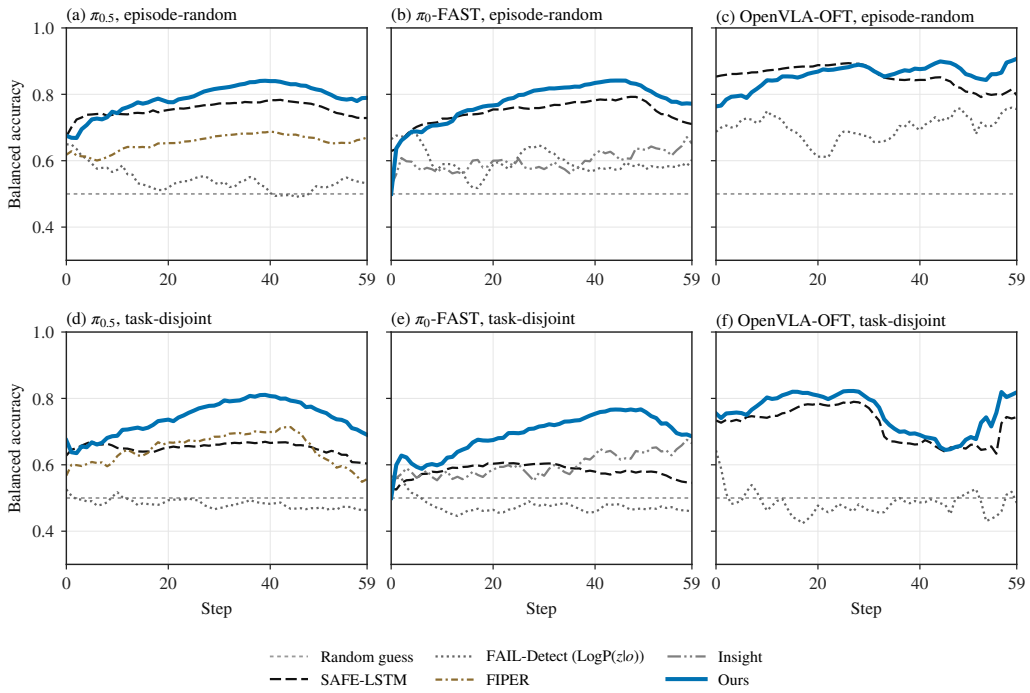

Figure 6: Per-step balanced accuracy is shown for episode-random (top) and task-disjoint (bottom) splits across three VLA families. Most baselines decline when evaluated on unseen tasks, whereas our method retains strong performance across the rollout.

C Ablation Survivor Features

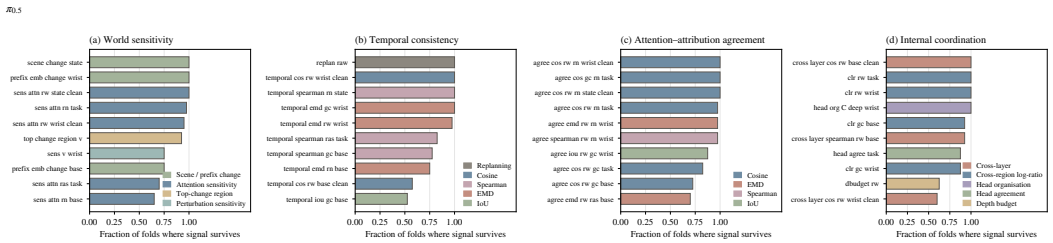

Figure 7: $\pi_{0.5}$: Complementary signals capture distinct aspects of failure.

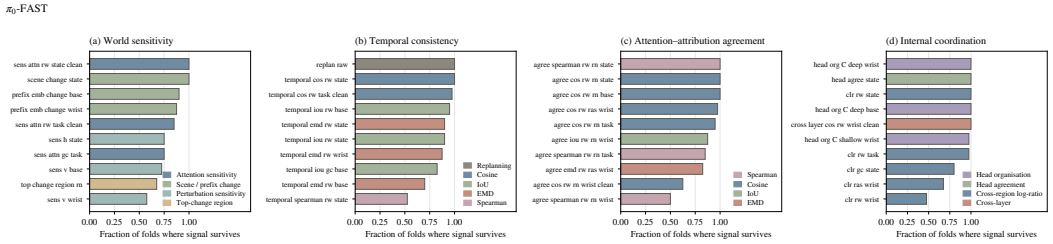

Figure 8: π_0-FAST: Complementary signals capture distinct aspects of failure.

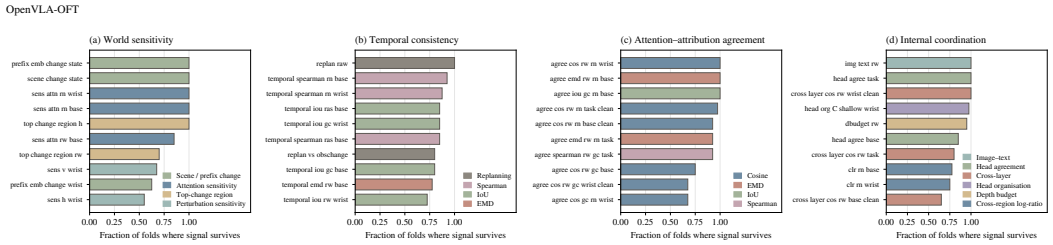

Figure 9: OpenVLA-OFT: Complementary signals capture distinct aspects of failure.

D ABLATION SURVIVOR PERFORMANCE

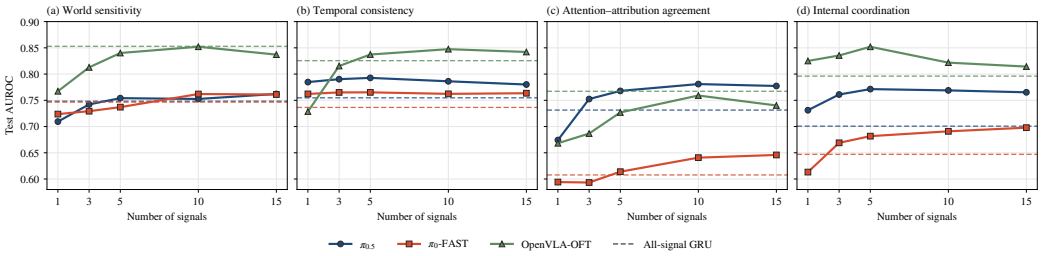

Figure 10: Prediction performance saturates with a small number of signals.