PREDICTING GENERATION FAILURES FROM INTERMEDIATE REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models based on diffusion or flow matching can produce undesirable, low-quality, or invalid outputs, which we broadly refer to as generation failures. Detecting such failures before the generation is complete can help identify bad samples early and avoid spending additional computation on samples that will eventually be discarded. In this work, we study whether generation failures can be forecast from intermediate representations of the generative model, and how early such signals become useful. We find that simple reconstruction-based models, including vanilla autoencoders and Block Sparse Featurizers (BSF), expose markers of eventual generation failure through their reconstruction residuals and feature usage patterns. On La-Proteina, a flow matching based protein generative model, these signals become predictive very early (first quarter) in generation and outperform the sequence-based baselines at the same stage. For low-entropy failures, we further find that the representation-level signal appears before the same failure becomes visible in the model's predicted sequence. We see a similar trend in image generation, where the reconstruction-based detectors perform comparably to or better than existing detectors on several datasets while 15-45% of the generation process still remains. Together, these results suggest that information about eventual generation failure can become accessible inside the model well before the final sample is produced, and in some cases before the failure is expressed in the generated output. More broadly, the recurrence of this behavior across protein and image generation suggests that reconstructing intermediate representations can serve as a simple, general tool for studying when generation failures become predictable.

1 INTRODUCTION

Diffusion (Ho et al., 2020) and Flow Matching (Lipman et al., 2022) models have rapidly emerged as widely used generative frameworks across a range of domains, including image and video synthesis (Ho et al., 2022; Khachatryan et al., 2023; Song et al., 2026), audio and speech generation, molecular and drug design, 3D generation, and protein sequence and structure generation (Geffner et al., 2026; Jing et al., 2023; Yim et al., 2023). This widespread adoption stems from their ability to produce high-quality samples while offering a favorable trade-off between computational cost and generation quality. These properties have enabled their use across diverse modalities and downstream applications. In this work, we focus on image and protein generation, two substantially different domains that allow us to study the same question across different output modalities, model architectures, and generative formulations.

Despite their success, these models can produce undesirable, low-quality, or invalid outputs, which we broadly refer to as generation failures Aithal et al. (2024); Ashiq et al. (2026); Tian et al. (2026); Geffner et al. (2026). In images, such failures can appear as structural inconsistencies such as malformed hands or unrealistic objects, while in protein generation they can appear as low-quality or degenerate sequences. These failures are especially problematic in applications where generated samples are passed through additional downstream validation or evaluation, since a failed sample may only be identified after the generation process has already completed.

Existing approaches identify generation failures using different signals from the generative process. Prior works have studied temporal instability during denoising (Aithal et al., 2024), local intrinsic

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.

dimension and score-based properties (Sobieski et al., 2026), anomalous regions (Kim et al., 2024), and changes in the learned vector field during generation (Tian et al., 2026). Other approaches rely on external models or domain-specific checks applied to the generated output. While these methods provide useful indicators of failure, the signals they use are often tied to the output space, score function, sampling trajectory, or a domain-specific verifier. This motivates us to ask *whether the intermediate representations of the generator already contain information about an eventual failure before it is visible in the output.*

We study whether generation failures can be forecast directly from intermediate representations before the generation is complete. To study this, we use simple reconstruction-based models, including standard autoencoders and Block Sparse Featurizers (BSFs) Fel et al. (2026), trained only on intermediate representations of the denoiser obtained from unlabeled generations. We use reconstruction residuals and feature usage patterns as markers of generation failure. These markers do not require access to a score function, stochastic sampling ensemble, or domain-specific verifier, allowing us to apply the same formulation across diffusion and flow matching models.

On La-Proteina Geffner et al. (2026), a recent flow matching based protein generative model, the intermediate representations become predictive very early in the generation and outperform the sequence-based baselines at the same stage. For low-entropy failures, we further find that the representation-level signal appears before the failure becomes visible in the predicted sequence, and these failures also concentrate their activation on a small shared set of BSF blocks or concepts (used interchangeably). We observe a similar trend across image datasets, where reconstruction-based detectors perform comparably to or better than existing detectors. Overall, these results show that information about an eventual generation failure can be present in the internal representations well before the final sample is produced, and in some cases before the failure becomes visible in the generated output.

In summary, our contributions are:

- We investigate whether generation failures can be forecast from internal representations before generation completes, and find that reconstruction-based signals can predict eventual failures well before the final sample is produced.
- On the protein generation task, these signals are predictive early across two La-Proteina variants as early as step 43 out of 400 denoising steps.
- We analyze the failure signal and find that low-entropy protein failures converge to a small shared set of BSF concepts, with the representation-level signal appearing before the same failure becomes visible in the generated sequence.
- Across image datasets, detectors built on intermediate representations match or outperform existing detectors while 15–45% of the generation process remains.

2 Related Work

Diffusion Models and Guidance: Diffusion models have emerged as state-of-the-art generative frameworks, relying on a forward noise-adding process and a learned reverse denoising process (Sohl-Dickstein et al. (2015); Ho et al. (2020). The reverse process is mathematically equivalent to score-matching (Hyvärinen (2005); Song et al. (2020), linking the denoiser predictions to the score function of the perturbed data distribution. To control generation, various guidance methods have been introduced. External classifier guidance (Dhariwal & Nichol (2021) uses gradients to steer the sampling trajectory toward a target class, while classifier-free guidance (Ho & Salimans (2022) samples from a combination of conditional and unconditional scores. Other lines of work have incorporated training-free guidance by imposing linear or non-linear constraints during sampling to improve fidelity in high-probability regions.

Understanding Hallucinations in Generative Models: While hallucinations have been extensively studied in Large Language Models and Large Vision-Language Models, they remain relatively underexplored in image-focused diffusion models. Prior literature frequently defines diffusion hallucinations as text-image misalignment, where the model fails to adhere to the conditioning prompt, leading to post-training alignment solutions. In contrast, recent studies argue that hallucinations are a fundamental property of the diffusion model's score function itself, independent of text condition-

ing. (Aithal et al. (2024) attribute these intrinsic hallucinations to mode interpolation, which is a phenomenon where the model generates artifacts by inappropriately interpolating between disjoint data modes due to smooth learned score approximations. Subsequent works have supported this by identifying vector field instabilities and temporal score anomalies (Cao et al. (2025) as primary drivers of out-of-distribution generation.

Mitigation Strategies and Limitations: Early attempts to address structural hallucinations were largely task-specific, focusing on filtering localized failure modes like deformed hands (Narasimhaswamy et al. (2024); Lu et al. (2024) or visual text generation bias (Lu et al. (2025). In conditional image translation and reconstruction, Kim et al. (2024) proposed local diffusion pipelines that process out of distribution regions separately. however, this approach requires expert mask annotations and is limited to image-conditional settings. To mitigate hallucinations in more general settings, several recent training-free interventions have been proposed, though they introduce distinct trade-offs. Adaptive Attention Modulation (Oorloff et al. (2025) scales the self-attention softmax temperature to suppress noise in early denoising stages but relies heavily on an external anomaly detector and access to original training data. Dynamic Guidance (Triaridis et al. (2025) selectively applies classifier guidance toward the most probable class at each step to sharpen hallucination-prone score directions, yet this necessitates an independently trained noisy classifier and has been observed to induce mode collapse. Alternatively, Tian et al. (2026) formulates diffusion as a continuation method to intrinsically intervene upon detecting vector field instabilities.

Protein Generation and Failure Detection: Generative models have increasingly been used for protein sequence and structure design, ranging from autoregressive sequence models such as ProGen Madani et al. (2023) and antibody language models such as p-IgGen Turnbull et al. (2024), to structure-conditioned sequence design and diffusion or flow-based generators. Recent approaches including ProteinMPNN Dauparas et al. (2022), RFdiffusion Watson et al. (2023), IgDiff Cutting et al. (2025), Chroma Ingraham et al. (2023), DPLM Wang et al. (2024), MultiFlow Campbell et al. (2024), and La-Proteina Geffner et al. (2026) generate protein sequences, structures, or jointly model both modalities. Generative models have also been used to construct antibody libraries with desirable developability properties, with generated sequences further validated using computational and experimental checks Rajagopal et al. (2025).

3 METHOD

3.1 TASK SETUP

We study how generative failures can be detected before the generation is complete. Consider a generative process that produces a final sample x^T using a sequence of T denoising steps. Let $\mathcal{D} = \{x_i^T\}_{i=1}^n$ be a set of generated samples. Each of the generated samples can be assigned a label indicating if the generation is a failure (hallucination, low quality, or invalid).

Our task is to predict if the generation will be a failure before the generation has finished. During an intermediate denoising timestep $t < T$, we extract an intermediate latent representation $H_t^{(\ell)}$ from a selected layer (ℓ) of the generative model (diffusion or flow matching models). We want to have a function that separates trajectories that eventually produce acceptable samples from those that produce samples flagged as failures: $s_t = f\left(H_t^{(\ell)}\right)$. Importantly, f does not have access to representations from future generation steps or to the completed final output at inference time.

3.2 LATENT RECONSTRUCTION

We explore several autoencoder and feature decomposition methods, with additional models (TopK SAEGao et al. (2025); Bussmann et al. (2024), and Matryoshka SAEs Bussmann et al. (2025); Zaigrajew et al. (2025)) described in the Appendix. Here, we focus on the two approaches that showed the most promising results, the Block-Sparse Featurizer (BSF) Fel et al. (2026), which imposes structured sparsity on the learned representation, and a standard autoencoder (AE), which provides an unconstrained reconstruction. Although the two approaches impose different constraints on their learned representations, both serve the purpose of modeling intermediate representations and providing quantities from which we derive markers of generation failure.

Given an intermediate representation $H_t^{(\ell)} \in \mathbb{R}^{P\times d}$ extracted from layer ℓ at generation step t, we treat each of the P positions as a d-dimensional embedding $e_i \in \mathbb{R}^d$, which serves as input to the BSF or AE. Here, P depends on the domain as for images, $P = h \times w$, where h and w are the spatial dimensions of the feature map, while for proteins, $P = r$, where r is the number of residues in the protein sequence. For image data, following prior work Surkov et al. (2025); Cywiński & Deja (2025), we view the intermediate representation $H_t^{(\ell)} \in \mathbb{R}^{h\times w\times d}$ as a collection of $h \times w$ spatial embeddings. Each spatial location is represented by an embedding $e_i \in \mathbb{R}^d$ capturing the activations across all feature channels at that location.

Block Sparse Featurizer: A BSF (Grassmannian variant) Fel et al. (2026) represents an embedding $e \in \mathbb{R}^d$ as a sparse sum of points from G learned subspaces. Its dictionary $\mathcal{F} = D_g{}_{g=1}^{G}$ consists of blocks $D_g \in \mathbb{R}^{d\times b}$ with orthonormal columns, each spanning a b-dimensional subspace, with $G = dE/b$ for an expansion factor E. Given an embedding e, the BSF computes block codes $z_g \in \mathbb{R}^b$ and a reconstruction $\hat{e} \in \mathbb{R}^d$: $z_g = \gamma D_g^\top e \qquad \hat{e} = \sum_{g\in\mathcal{S}_K(e)} D_g z_g$. Here γ is a learned scalar, and $\mathcal{S}_K(e)$ is the set of the K blocks with the largest code norm $\|z_g\|_2$, the codes of all other blocks are set to zero. The encoder and decoder share the blocks, so the same D_g both reads the code and reconstructs from it.

Autoencoder: As a model without a feature dictionary or sparsity constraint, an AE maps the embedding through a bottleneck of width $m < d$: $z = f_{\text{enc}}(e) \in \mathbb{R}^m \qquad \hat{e} = f_{\text{dec}}(z)$, where f_{enc} and f_{dec} are MLPs. Both models are trained on embeddings from unlabeled generations to minimize the reconstruction error $\|e - \hat{e}\|_2^2$.

3.3 Markers of Generation Failures

We find two markers to be particularly effective for detecting generation failures, namely the reconstruction residual and negative feature entropy. The reconstruction residual measures how well the intermediate embeddings can be reconstructed by the learned autoencoder, while the feature entropy measures how broadly the learned features are used across a sample.

Given an intermediate representation $H_t^{(\ell)}$, let $\{e_i\}_{i=1}^P$ denote its P embeddings and $\{\hat{e}_i\}_{i=1}^P$ their corresponding reconstructions obtained using the BSF or AE. We denote the bottleneck representation of each embedding by Z_i. For BSF, $Z_i = [z_{1,i}, \ldots, z_{G,i}]^\top \in \mathbb{R}^{G\times b}$, where G is the number of feature blocks, b is the dimension of each block, and $z_{g,i} \in \mathbb{R}^b$ is zero for unselected blocks. Across all P positions in a sample, the resulting feature representation is $\mathcal{Z} \in \mathbb{R}^{P\times G\times b}$. For AE, the bottleneck representation is $Z_i \in \mathbb{R}^m$, $\mathcal{Z} \in \mathbb{R}^{P\times m}$, where m is the bottleneck dimension. We use the reconstruction pairs $(e_i, \hat{e}_i)$ to define the reconstruction residual and use the feature representation $\mathcal{Z}$ to define the feature entropy.

Reconstruction Residual: For each spatial embedding, we measure its squared reconstruction error relative to its squared magnitude. Averaging over positions gives the sample-level residual: $s_{\text{R}}\left(H_t^{(\ell)}\right) = {}^1\!/_P \sum_{i=1}^P {}^{\|e_i-\hat{e}_i\|_2^2}\!/_{\|e_i\|_2^2}$. Normalizing the residuals allows us to combine and compare them across positions with different activation scales. We use this same formulation of residual for both BSF and AE.

Negative Feature Entropy: We additionally measure how broadly the learned dictionary blocks (BSF) or bottleneck features (AE) are used across the sample. Let $z_{g,i}$ denote the code of block g for embedding e_i (for AE, each of the $G{=}m$ bottleneck units is a block and all are retained). We first sum the magnitudes of its retained codes (Top K) across spatial positions:

$$a_g = \sum_{i=1}^{P} \mathbf{1}[g \in \mathcal{S}_K(e_i)]\, \|z_{g,i}\|_2, \qquad p_g = \frac{a_g}{\sum_{r=1}^{G} a_r}. \tag{1}$$

Thus, p_g is the fraction of the sample's total retained code magnitude assigned to block g. Using this distribution, we define negative feature entropy as $s_{\text{NFE}}\left(H_t^{(\ell)}\right) = -\mathcal{H}(p) = \sum_{g=1}^{G} p_g \log p_g$. A score closer to zero indicates activity concentrated in fewer dictionary blocks and the intermediate representation is using fewer features, whereas a more negative score indicates activity spread across more blocks.

Equal-weight fusion: We also explore combining the reconstruction residual and negative feature entropy. The direction in which each marker indicates failure may differ across generators and denoising steps, so we fix it on the validation set, for $q \in \{\mathrm{R}, \mathrm{NFE}\}$ we set $\sigma_q = +1$ if its validation ROC-AUC is at least 0.5 and $\sigma_q = -1$ otherwise, and use the signed marker $\tilde{s}_q\Big(H_t^{(\ell)}\Big) = \sigma_q\, s_q\Big(H_t^{(\ell)}\Big)$. To account for their different scales, we map each marker to its rank relative to the n validation representations at the same layer and step:

$$r_q\Big(H_t^{(\ell)}\Big) = \frac{1}{n+1}\sum_{j=1}^{n} \mathbf{1}\Big[\tilde{s}_q\Big(H_{t,j}^{(\ell)}\Big) \le \tilde{s}_q\Big(H_t^{(\ell)}\Big)\Big], \quad q \in \{\mathrm{R}, \mathrm{NFE}\}. \tag{2}$$

The fused score is their equal-weight average: $s_{\mathrm{fused}}\Big(H_t^{(\ell)}\Big) = \Big(r_{\mathrm{R}}\Big(H_t^{(\ell)}\Big) + r_{\mathrm{NFE}}\Big(H_t^{(\ell)}\Big)\Big)/2$. The signs σ_q and the validation reference distributions are fixed before test evaluation.

4 Experiments

Generative models are widely used across applications including natural image synthesis Aithal et al. (2024), medical imaging Tivnan et al. (2024), materials design Yang et al. (2024); Levy et al. (2025), and protein sequence and structure generation Watson et al. (2023); Geffner et al. (2026), where generated outputs are often studied for hallucinations, structural inconsistencies, physical invalidity, or other forms of generation failure. In this work, we chose two well-studied and diverse domains from this as our choice of applications, and future work includes a detailed study on more generalized and a variety of applications

We conduct experiments across protein sequence and image generation domains to understand if we can forecast generation failures during generation, how early we can predict them, and what is best suited to detect them. We generate distinct samples and split them into train, validation, and test sets. The validation and test sets are used for the evaluation of the detectors and are given binary labels to differentiate between good and failed generations. The labeling process for each domain is detailed in its respective subsection. Train sets are unlabeled and used for training the BSF and AE (details in the appendix). In line with prior work, we use ROC-AUC, PR-AUC, and Cohen's d to evaluate the detection performance. For all the detectors, we find the best-performing hyperparameters like dictionary size, K, bottleneck size, and the best marker (s_{R}, s_{NFE}, s_{fused}) on the validation set and report results with that setting on the test set.

4.1 Protein Sequence Generation

4.1.1 Setup and Baselines

We generate protein sequences using two variants of La-Proteina Geffner et al. (2026), LD1 and LD2, with 400 generation steps, and evaluate whether we can detect bad sequences early on. With LD1 we generate 300-residue-long sequences and with LD2 500-residue-long sequences. We flag a sequence as a failed generation or bad sequence if it has high perplexity or low entropy, with the labeling criteria detailed in the Appendix. For LD1, we generate 25,000 training, 5,000 validation, and 15,000 test samples, and for LD2, 10,000 training, 2,500 validation, and 7,500 test samples. For our detectors, we use the layer-9 denoiser representations and train BSF and AE on the unlabeled training set, with all configurations (K, timestep) selected on validation data.

We use Protein Language Models (PLMs) ESM-2 (650M and 8M) Lin et al. (2022) and the language model (LLM) RoBERTa Liu et al. (2019) as baselines, using their mean per-residue negative log-likelihood as the detection score. At intermediate timesteps, we evaluate the baselines on La-Proteina's clean sequence prediction available at that step. We use timestep 100 as our main detection point for LD1 and timestep 43 for LD2. Further details on labeling, detector configurations, and baseline evaluation are provided in the Appendix.

4.1.2 Detection Performance

The results for protein sequence failure detection are shown in Table 1. For LD1, at denoising timestep 100 of 400, both reconstruction-based models improve over the language model baselines

Table 1: Generation failure detection on two La-Proteina variants. LD1 uses sequence length 300 and LD2 uses sequence length 500. All methods are evaluated at the corresponding early detection checkpoint (timestep 100 for LD1 and 43 for LD2). d is Cohen's d between failed and good proteins. **Bold** marks the best result per column; underline marks the second best. The row marked Δ gives the percentage improvement of our best variant (AE or BSF) over the best baseline (ESM-2-8M / ESM-2-650M / RoBERTa).

Variant	Method	Low-Entropy			High-Perplexity			Combined		
		PR-AUC	ROC-AUC	d	PR-AUC	ROC-AUC	d	PR-AUC	ROC-AUC	d
LD1 $N = 300$	ESM-2-8M	0.031	0.566	0.237	0.052	0.540	0.170	0.079	0.548	0.192
	ESM-2-650M	0.050	0.627	0.476	0.048	0.501	0.019	0.084	0.542	0.168
	RoBERTa	0.021	0.384	−0.286	0.075	0.649	0.455	0.089	0.563	0.213
	AE (fused)	0.044	0.698	0.670	0.082	0.693	0.644	0.118	0.695	0.656
	BSF (fused)	**0.283**	**0.887**	**1.428**	**0.094**	**0.738**	**0.768**	**0.232**	**0.786**	**0.990**
	Δ *vs. baseline*	+466%	+41%	+200%	+25%	+14%	+69%	+161%	+40%	+365%
LD2 $N = 500$	ESM-2-8M	0.099	0.615	0.463	0.029	0.535	0.115	0.115	0.589	0.348
	ESM-2-650M	0.104	0.657	0.516	0.032	0.580	0.285	0.124	0.631	0.442
	RoBERTa	0.062	0.438	−0.070	**0.041**	0.594	0.364	0.093	0.490	0.072
	AE (fused)	0.681	**0.927**	**1.728**	0.034	0.618	0.386	0.519	0.825	**1.272**
	BSF (fused)	**0.732**	0.906	1.540	0.040	**0.680**	**0.612**	**0.561**	**0.831**	1.232
	Δ *vs. baseline*	+604%	+41%	+235%	–	+14%	+68%	+352%	+32%	+188%

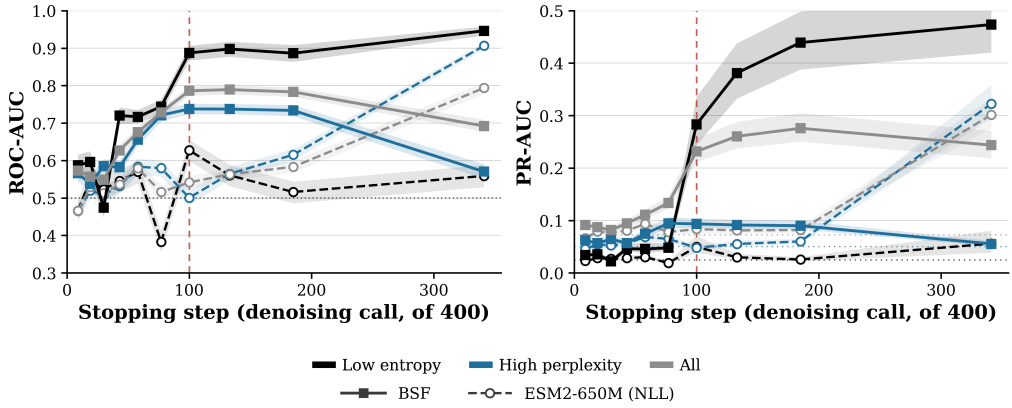


Figure 1: Detection performance across different generation timesteps for BSF and ESM-2-650M based detectors (LD1 variant) . BSF operates on the intermediate denoiser representations, while ESM-2-650M scores the clean sequence prediction available at each timestep.

on the combined (low-entropy and high-perplexity) failure detection task. BSF gives the best overall performance with a ROC-AUC of 0.786, compared to 0.563 for the best baseline, while AE achieves a ROC-AUC of 0.695. We observe a similar improvement for the two failure types separately, with BSF achieving a ROC-AUC of 0.887 for low-entropy and 0.738 for high-perplexity sequences, compared to 0.627 and 0.649 for the best baselines. A similar trend can also be seen with PR-AUC and Cohen's d. For LD2, the signal is available even earlier, at timestep 43 of 400, where BSF and AE achieve combined ROC-AUCs of 0.831 and 0.825, compared to 0.631 for the best baseline, with similar gains for both failure types. Importantly, all the methods in this comparison use information available at the same generation timestep. These results show that the intermediate representations of La-Proteina already contain useful information about the quality of the eventual generated sequence after only a **quarter** of the generation process for LD1, and about a **tenth** for LD2.

4.1.3 Detection Across Generation Timesteps

We next evaluate the reconstruction-based detector and the PLM baseline across different generation timesteps to understand how early the failure signal becomes useful in LD1 (LD2 results in Appendix). The results are shown in Figure 1. We observe that BSF starts separating good and failed generations by timestep 100, with only 25% of the generation completed. The behavior also varies across the two types of failures. BSF detects low-entropy sequences particularly well from timestep 100 onward, while also providing stronger detection of high-perplexity sequences during

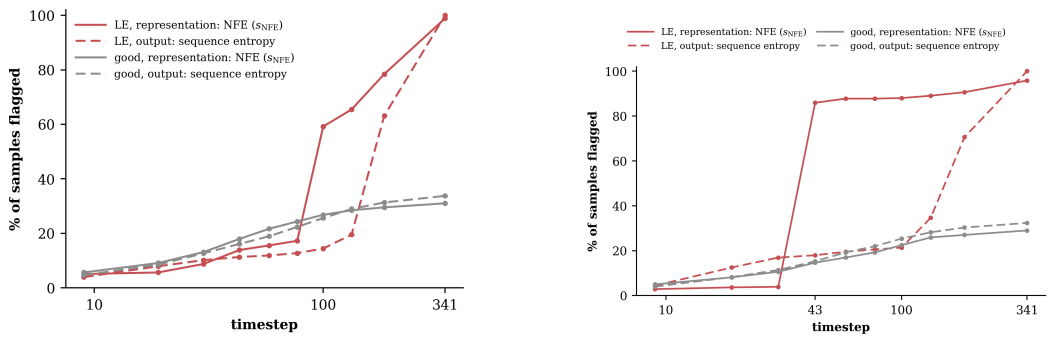


Figure 2: Percentage of Low-Entropy (LE) sequences flagged over generation by NFE (solid) and predicted-sequence entropy (dashed) for La-Proteina LD1 (**left**) and LD2 (**right**), using thresholds from good validation sequences. NFE flags many low-entropy failures by timestep 100 in LD1 and by timestep 43 in LD2, while sequence entropy rises only around timestep 185 in both.

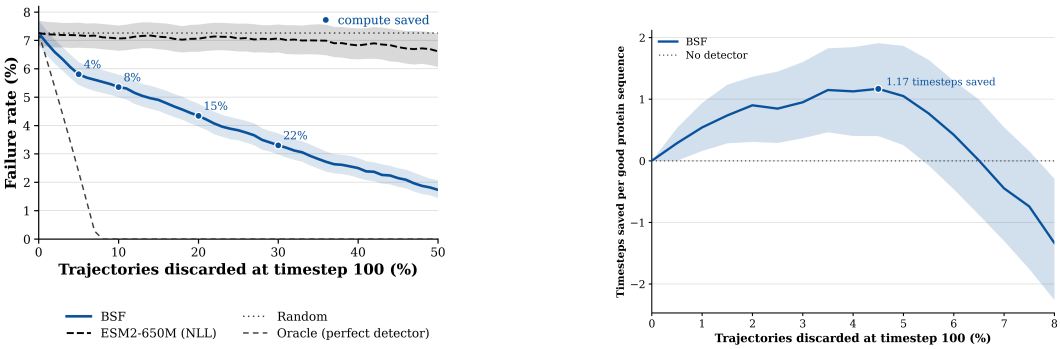


Figure 3: **Left:** Failure rate versus fraction of detector-flagged trajectories discarded. Random selection leaves the failure rate unchanged, while the oracle represents a perfect detector. Annotations show the corresponding denoising compute saved. **Right:** Denoising timesteps saved per successful protein relative to generation without a detector.

the intermediate stages of generation. Towards the end of generation, the ESM-2 baseline improves substantially for high-perplexity sequences, as the generated sequence becomes closer to its final form. Overall, the results show that intermediate representations provide a useful failure signal much earlier in the generation process, when the sequence-based baseline still has limited information about the final output.

Practical utility of early detection: We next study if the detector can be used to make generation more efficient by stopping trajectories that are likely to fail. At timestep 100, we rank the trajectories using the BSF score, discard the highest scoring fraction, and continue generation only for the remaining samples. As shown in Figure 3, this gives a clear trade-off between output quality and compute. For example, discarding the top 20% of trajectories reduces the failure rate from 7.26% to 4.34% while saving about 15% of the denoising computation. At an operating point of around 4-5% discard, the failure rate is reduced to about 5.8% while also reducing the compute required per successfully generated protein with 1.17 timesteps saved per good generation. These results show that the early failure signal is not only predictive, but can also be used as a practical screening mechanism to improve the quality-compute trade-off during generation.

4.1.4 Understanding the Failure Signal

We next analyze what the BSF concepts are capturing and how early these signals appear during generation. We study two questions, whether different failure types activate distinct sets of concepts, and whether the representation-level signal appears before the same failure becomes visible in the generated sequence (predicted or final). Further details on concept selection and thresholding are provided in the Appendix.

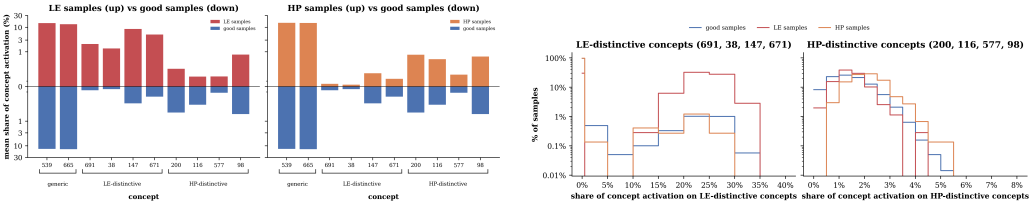


Figure 4: **Left:** Mean share of BSF activation per concept at timestep 100 for failed (up) and good (down) sequences. LE-distinctive concepts are rarely used by good sequences, while the generic concepts are shared by all. **Right:** Share of each sequence's activation on the four LE-distinctive (left) and HP-distinctive (right) concepts. Around 70% of low-entropy sequences place 15-35% of their activation on the LE-distinctive concepts, whereas good sequences place almost none. High-perplexity sequences use their distinctive concepts only slightly more than good sequences.

Low-entropy failures concentrate on a small shared set of concepts: At timestep 100, we measure the fraction of total BSF activation assigned to each concept and identify concepts that are more active in low-entropy (LE) or high-perplexity (HP) sequences than in good sequences (distinctive concepts). As shown in Figure 4, most LE failures place substantially more activation on a small set of four concepts, while these concepts are rarely used by good sequences. On average, an LE sequence places 17% of its activation on these concepts compared to 0.6% for a good sequence, and around 70% of LE sequences activate all four together. These concepts alone separate LE from good sequences with a ROC-AUC of 0.84. We also find that the five most over-used concepts of each sequence are much more similar across LE sequences, with a mean pairwise Jaccard similarity of 0.38 compared to 0.03 for matched good sequences. In contrast, HP failures do not show a similarly strong shared pattern, as their distinctive concepts receive only a small increase in activation and their overlap remains low. This suggests that LE failures converge to a common concept-level failure mode, while HP failures are more diffuse and sample-dependent.

The representation-level signal appears before the failure is visible in the output: We next track the BSF negative feature entropy (NFE) and the entropy of La-Proteina's predicted sequence across generation timesteps. For each signal, we flag a sample once it crosses a threshold selected using good validation samples. Figure 2 shows that the representation-based signal becomes informative substantially earlier than the output. At this stage, the predicted sequences have not yet developed the low-entropy behavior that defines the final failure. The output-based signal only catches up later, around timestep 185, when the predicted sequence begins to collapse. We do not observe the same clear lead for HP failures. These results indicate that, for LE failures, the BSF signal is not simply reading the failure from the current output; the failure signal is already present in the internal representation before it becomes visible in the generated sequence.

4.2 Image Generation

Setup and Baselines: We use five image datasets used in prior work Sobieski et al. (2026) and evaluate how early we can predict failures. For synthetic images, we use SimpleShapes Aithal et al. (2024) and MNIST LeCun et al. (1998). For natural images, we evaluate on AFHQv2 Choi et al. (2020) with animal faces, FFHQ Karras et al. (2019) with human faces, and 11kHands Afifi (2019) with hand images. Images are at 64×64 resolution for AFHQv2 and FFHQ, and 128×128 for 11kHands. We use the same models and setup as in Sobieski et al. (2026). For SimpleShapes, we use the DDPM Ho et al. (2020) based UNet, while for MNIST we use the pretrained HuggingFace UNet2 model, both use 250-step sampling for inference. For AFHQv2 and FFHQ, we use the unconditional VE-EDM checkpoints released with Karras et al. (2022), and for 11kHands we use the pretrained VE-EDM model from Tian et al. (2026), all with deterministic 40-step Euler sampling. We compare against prior detectors Sobieski et al. (2026); Aithal et al. (2024) where **LID/LMI** Sobieski et al. (2026) score images via denoising score matching loss and variance of regenerated perturbed samples and are applied to the final sample. **TVF** Aithal et al. (2024) measures temporal variance of predicted clean images over a trajectory window. The details of the data split are in the appendix. LID and TVF hyperparameters are selected via validation ROC-AUC. While LID and LMI are applied to completed samples in our standard baseline evaluation, TVF can be evaluated as soon as its selected trajectory window ends. Following the same labeling protocol as Sobieski et al. (2026), for MNIST, AFHQv2, FFHQ, and 11kHands/Hands, two human annotators label the final generated images as successful (correct) , failed (hallucination), or uncertain (ambiguous). We

exclude uncertain labels separately for each annotator, compute the evaluation metrics using their respective labels, and report the mean across annotators (details in Appendix). We select detector configurations and denoising steps using validation ROC-AUC among the considered candidates (details in Appendix). These choices are then fixed, and we report ROC-AUC, PR-AUC, and Cohen's d on the test split.

Early Failure Detection Performance: The results across the image datasets are shown in Tables 2 and 3. We find that reconstruction-based detectors can identify generation failures early, while remaining comparable to or improving over post-generation baselines. The improvement with reconstruction-based methods is larger for the more subtle failures in AFHQv2 and FFHQ. For more obvious pixel-space failures such as Hands and SimpleShapes, LID performs better, while reconstruction-detectors remain within 0.03 ROC-AUC. On AFHQv2, AE achieves a top ROC-AUC of 0.728 with 22.5% of denoising remaining (vs. 0.614 for the best baseline). On FFHQ, BSF reaches a 0.757 ROC-AUC with 45% remaining, beating the 0.620 baseline. Similar trends hold for synthetic datasets where AE achieves 0.713 ROC-AUC on MNIST with 28.4% remaining, while BSF matches the best 0.824 baseline on SimpleShapes with 15.2% remaining. Overall, intermediate representations provide vital, early failure signals, though the optimal model (AE vs. BSF) varies by dataset.

	MNIST				Simple Shapes			
Method	PR-AUC ↑	ROC-AUC ↑	d ↑	Steps left (%)↑	PR-AUC ↑	ROC-AUC ↑	d ↑	Steps left (%)↑
TVF	0.394	0.629	0.513	**31.2**	0.584	<u>0.746</u>	1.171	6.4
LMI	0.379	0.589	0.389	0	0.332	0.642	0.540	0
LID	<u>0.432</u>	**0.724**	<u>0.774</u>	0	**0.741**	**0.824**	**1.987**	0
BSF	0.350	0.625	0.437	<u>28.4</u>	<u>0.670</u>	**0.824**	<u>1.529</u>	<u>15.2</u>
AE	**0.463**	<u>0.713</u>	**0.822**	<u>28.4</u>	0.406	0.442	0.396	**28.4**

Table 2: Comparison of generation failure detectors on MNIST and Simple Shapes. Methods report test performance, with configurations selected by validation ROC-AUC among the reported candidates. BSF uses residual-entropy fusion for MNIST and residual alone for Simple Shapes. AE uses reconstruction residual throughout. LID and LMI evaluate completed generations while TVF uses a selected denoising trajectory window. Steps left denotes the percentage of sampler calls remaining at detection. (d is Cohen's d). **Bold** marks the best result per column; <u>underline</u> marks the second best.

Table 3: Comparison of generation failure detectors across three image datasets. Methods report test performance, with configurations selected by validation ROC-AUC among the reported candidates. BSF uses residual-entropy fusion for AFHQv2 and residual alone for FFHQ and Hands. AE uses reconstruction residual throughout. LID and LMI evaluate completed generations while TVF uses a selected denoising trajectory window. Steps left denotes the percentage of sampler calls remaining at detection. (d is Cohen's d). **Bold** marks the best result per column; <u>underline</u> marks the second best.

	AFHQv2				FFHQ				Hands			
Method	PR-AUC ↑	ROC-AUC ↑	d ↑	Steps left (%)↑	PR-AUC ↑	ROC-AUC ↑	d ↑	Steps left (%)↑	PR-AUC ↑	ROC-AUC ↑	d ↑	Steps left (%)↑
TVF	0.206	0.612	0.493	<u>35</u>	0.147	0.620	0.342	<u>35</u>	0.497	0.782	0.980	25
LMI	0.155	0.614	0.054	0	0.179	0.585	0.406	0	0.405	0.647	0.447	0
LID	0.192	0.576	0.366	0	0.143	0.605	0.309	0	**0.704**	**0.847**	**1.472**	0
BSF	<u>0.212</u>	<u>0.629</u>	<u>0.499</u>	**45**	**0.267**	**0.757**	**0.956**	**45**	0.601	0.804	1.201	**37.5**
AE	**0.262**	**0.728**	**0.848**	22.5	<u>0.207</u>	<u>0.717</u>	<u>0.749</u>	**45**	<u>0.637</u>	<u>0.820</u>	<u>1.273</u>	<u>30</u>

5 Conclusion

In this work, we studied whether generation failures can be predicted from intermediate representations before the final sample is produced. Using reconstruction-based models, including Block Sparse Featurizers and standard autoencoders, we show that reconstruction residuals and feature usage patterns provide useful markers of eventual failure across protein and image generation. On La-Proteina, these signals become predictive early in generation, outperform sequence-based baselines at the same stage, and, for low-entropy failures, can appear before the failure is visible in the predicted sequence. We also show that early screening can reduce the failure rate while saving denoising compute. Across image datasets, the same approach remains competitive with or outperforms existing detectors while a substantial fraction of generation is still remaining. Overall, our results suggest that information about eventual generation failure can become accessible inside the model well before the final output is produced, and that reconstruction of intermediate representations provides a simple and general way to expose these signals.

Reproducibility Statement

We provide detailed experimental and implementation settings in the Appendix to facilitate reproducibility. For protein sequence generation, we report the La-Proteina checkpoints and generation settings, data splits, labeling criteria, baseline implementations, reconstruction model training details, hyperparameters, and validation-based configuration selection. For image generation, we provide the sampling settings, data splits and labeling protocol, baseline implementations, reconstruction model training details, and the validation sweeps used to select the detector configuration. All model and detector configurations are selected using the validation sets and are fixed before evaluation on the test sets. We also report the additional reconstruction models and ablation studies used to justify the design choices in our experiments.

References

Mahmoud Afifi. 11k hands: Gender recognition and biometric identification using a large dataset of hand images. *Multimedia Tools and Applications*, 78(15):20835–20854, 2019.

Sumukh K Aithal, Pratyush Maini, Zachary C Lipton, and J Zico Kolter. Understanding hallucinations in diffusion models through mode interpolation. *Advances in neural information processing systems*, 37:134614–134644, 2024.

Muhammad H Ashiq, Samanyu Arora, Abhinav N Harish, Ishaan Kharbanda, Hung Yun Tseng, and Grigorios G Chrysos. Why ddim hallucinates more than ddpm: A theoretical analysis of reverse dynamics. *arXiv preprint arXiv:2605.06831*, 2026.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. *arXiv preprint arXiv:2503.17547*, 2025.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint arXiv:2402.04997*, 2024.

Yu Cao, Zengqun Zhao, Ioannis Patras, and Shaogang Gong. Temporal score analysis for understanding and correcting diffusion artifacts. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7707–7716. IEEE, 2025.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8185–8194. IEEE, 2020.

Daniel Cutting, Frédéric A Dreyer, David Errington, Constantin Schneider, and Charlotte M Deane. De novo antibody design with se (3) diffusion. *Journal of Computational Biology*, 32(4):351–361, 2025.

Bartosz Cywiński and Kamil Deja. Saeuron: Interpretable concept unlearning in diffusion models with sparse autoencoders. In *Forty-second International Conference on Machine Learning*, 2025.

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, DEBSINDHU BHOWMIK, and Burkhard Rost. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *bioRxiv*, 2020. doi: 10.1101/2020.07.12.199554. URL https://www.biorxiv.org/content/early/2020/07/21/2020.07.12.199554.

Thomas Fel, Matthew Kowal, Mozes Jacobs, Dron Hazra, Usha Bhalla, Lee Sharkey, Lucius Bushnaq, Satchel Grant, Tal Haklay, Thomas Icard, et al. Structuring sparsity: Block-sparse featurizers capture visual concept manifolds. *arXiv preprint arXiv:2606.25234*, 2026.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, volume 2025, pp. 26721–26754, 2025.

Tomas Geffner, Kieran Didi, Zhonglin Cao, Danny Reidenbach, Zuobai Zhang, Christian Dallago, Emine Kucukbenli, Karsten Kreis, and Arash Vahdat. La-proteina: Atomistic protein generation via partially latent flow matching. In *International Conference on Learning Representations*, volume 2026, pp. 67350–67395, 2026.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6:695–709, 2005. URL https://api.semanticscholar.org/CorpusID:1152227.

John B Ingraham, Max Baranov, Zak Costello, Karl W Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M Lord, Christopher Ng-Thow-Hing, Erik R Van Vlack, et al. Illuminating protein space with a programmable generative model. *Nature*, 623(7989):1070–1078, 2023.

Bowen Jing, Ezra Erives, Peter Pao-Huang, Gabriele Corso, Bonnie Berger, and Tommi Jaakkola. Eigenfold: Generative protein structure prediction with diffusion models. *arXiv preprint arXiv:2304.02198*, 2023.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 4396–4405. IEEE, 2019.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Levon Khachatryan, Andranik Movsisyan, Vahram Tadevosyan, Roberto Henschel, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Text2video-zero: Text-to-image diffusion models are zero-shot video generators. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15908–15918. IEEE, 2023.

Seunghoi Kim, Chen Jin, Tom Diethe, Matteo Figini, Henry FJ Tregidgo, Asher Mullokandov, Philip Teare, and Daniel C Alexander. Tackling structural hallucination in image translation with local diffusion. In *European conference on computer vision*, pp. 87–103. Springer, 2024.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Daniel Levy, Siba Smarak Panigrahi, Sékou-Oumar Kaba, Qiang Zhu, Kin Long Kelvin Lee, Mikhail Galkin, Santiago Miret, and Siamak Ravanbakhsh. Symmcd: symmetry-preserving crystal generation with diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 23166–23189, 2025.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Rui Lu, Runzhe Wang, Kaifeng Lyu, Xitai Jiang, Gao Huang, and Mengdi Wang. Towards understanding text hallucination of diffusion models via local generation bias. In *International Conference on Learning Representations*, volume 2025, pp. 24192–24219, 2025.

Wenquan Lu, Yufei Xu, Jing Zhang, Chaoyue Wang, and Dacheng Tao. Handrefiner: Refining malformed hands in generated images by diffusion-based conditional inpainting. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 7085–7093, 2024.

Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41 (8):1099–1106, 2023.

Supreeth Narasimhaswamy, Uttaran Bhattacharya, Xiang Chen, Ishita Dasgupta, Saayan Mitra, and Minh Hoai. Handiffuser: Text-to-image generation with realistic hand appearances. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2468–2479. IEEE, 2024.

Trevine Oorloff, Yaser Yacoob, and Abhinav Shrivastava. Mitigating hallucinations in diffusion models through adaptive attention modulation. *arXiv preprint arXiv:2502.16872*, 2025.

Nandhini Rajagopal, Udit Choudhary, Kenny Tsang, Kyle P Martin, Murat Karadag, Hsin-Ting Chen, Na-Young Kwon, Joseph Mozdzierz, Alexander M Horspool, Li Li, et al. Deep learning-based design and experimental validation of a medicine-like human antibody library. *Briefings in Bioinformatics*, 26(1):bbaf023, 2025.

Bartlomiej Sobieski, Matthew Tivnan, Dawid Płudowski, Michał Jan Włodarczyk, Pengfei Jin, Przemyslaw Biecek, and Quanzheng Li. Local intrinsic dimension unveils hallucinations in diffusion models. *arXiv preprint arXiv:2605.05026*, 2026.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Kaiyu Song, Baoshun Tong, and Hanjiang Lai. Rethinking oversaturation in classifier-free guidance via low frequency. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 10107–10111. IEEE, 2026.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Viacheslav Surkov, Chris Wendler, Antonio Mari, Mikhail Terekhov, Justin Deschenaux, Robert West, Caglar Gulcehre, and David Bau. One-step is enough: Sparse autoencoders for text-to-image diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=MBJJ9Wcpg9.

Yiqi Tian, Pengfei Jin, Mingze Yuan, Na Li, Bo Zeng, and Quanzheng Li. Rods: Robust optimization inspired diffusion sampling for detecting and reducing hallucination in generative models. *Advances in Neural Information Processing Systems*, 38:25782–25814, 2026.

Matthew Tivnan, Siyeop Yoon, Zhennong Chen, Xiang Li, Dufan Wu, and Quanzheng Li. Hallucination index: An image quality metric for generative reconstruction models. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 449–458. Springer, 2024.

Kostas Triaridis, Alexandros Graikos, Aggelina Chatziagapi, Grigorios G Chrysos, and Dimitris Samaras. Mitigating diffusion model hallucinations with dynamic guidance. *arXiv preprint arXiv:2510.05356*, 2025.

Oliver M Turnbull, Dino Oglic, Rebecca Croasdale-Wood, and Charlotte M Deane. p-iggen: a paired antibody generative language model. *Bioinformatics*, 40(11):btae659, 2024.

Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. *arXiv preprint arXiv:2402.18567*, 2024.

Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

Sherry Yang, KwangHwan Cho, Amil Merchant, Pieter Abbeel, Dale Schuurmans, Igor Mordatch, and Ekin Cubuk. Scalable diffusion for materials generation. In *International Conference on Learning Representations*, volume 2024, pp. 47319–47334, 2024.

Jason Yim, Brian L Trippe, Valentin De Bortoli, Emile Mathieu, Arnaud Doucet, Regina Barzilay, and Tommi Jaakkola. Se (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*, 2023.

Vladimir Zaigrajew, Hubert Baniecki, and Przemyslaw Biecek. Interpreting clip with hierarchical sparse autoencoders. *arXiv preprint arXiv:2502.20578*, 2025.

AI USE STATEMENT

In this work, we used generative AI tools for paper writing, software code development, and brainstorming. We did not use generative AI tools to generate synthetic datasets, formulate mathematical claims, assist in writing proofs, or interpret experimental results. Other required disclosure tasks do not apply to this work. Additionally, we used generative AI tools to edit and refine the paper for readability, assist with software code, brainstorm research directions, and help organize parts of the manuscript. All AI-assisted text and code were reviewed by the authors. AI-assisted code was inspected and tested before being used in experiments, and AI-assisted writing was checked for technical correctness and consistency with the experiments and results. The authors made the final decisions on the research methodology, experiments, analysis, and claims presented in the paper. We take responsibility for the final content of this work, including text, claims, software, or other artifacts produced with the aid of generative AI.

A Protein Sequence Generation

A.1 Additional Results

We provide additional detection results across generation timesteps, comparing BSF with the sequence-based baselines for both La-Proteina variants (Figures 5, 6, 7).

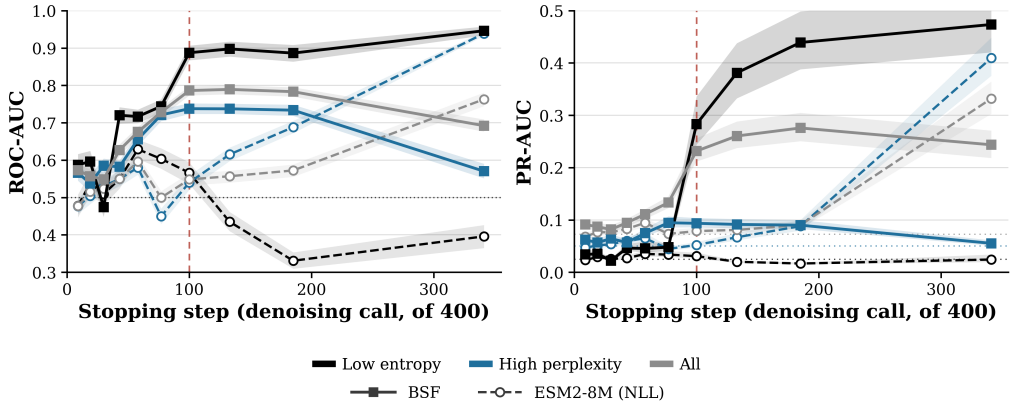


Figure 5: Detection performance across different generation timesteps for BSF and ESM-2-8M based detectors (LD1 variant). We report ROC-AUC (left) and PR-AUC (right). BSF operates on the intermediate denoiser representations, while ESM-2-8M scores the clean sequence prediction available at each timestep.

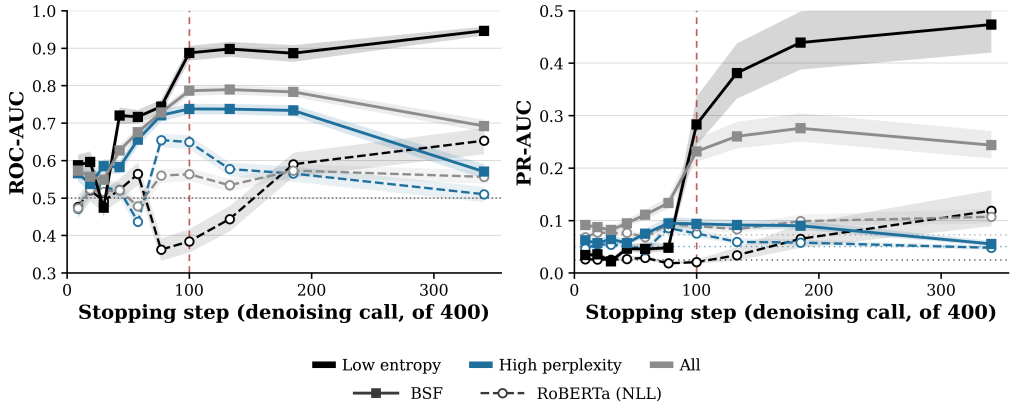


Figure 6: Detection performance across different generation timesteps for BSF and RoBERTa-based detectors (LD1 variant). We report ROC-AUC (left) and PR-AUC (right). BSF operates on the intermediate denoiser representations, while RoBERTa scores the clean sequence prediction available at each timestep.

A.2 Generation Setup

We generate protein sequences using two unconditional variants of La-Proteina Geffner et al. (2026), with the released checkpoints and the default unconditional inference configuration of 400 generation steps. The variant referred to as LD1 uses the `LD1_ucond_notri_512` checkpoint, which does not use triangular updates. LD2 uses the `LD2_ucond_tri_512` checkpoint, which includes triangular updates. We generate 300 residue sequences with LD1 and 500 residue sequences with LD2. For LD1, we generate 25,000 training proteins and 20,000 evaluation protein sequences, with the evaluation set split into 5,000 validation and 15,000 test proteins. For LD2, we generate 10,000 training proteins and 10,000 evaluation proteins, which are split into 2,500 validation and 7,500 test proteins. We ensure that there are no repeated protein sequences among all the sets.

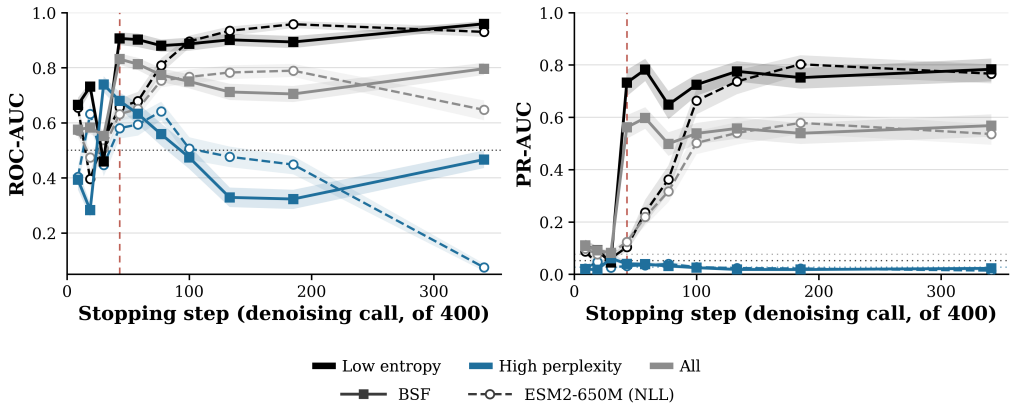


Figure 7: Detection performance across different generation timesteps for BSF and ESM-2-650M based detectors (LD2 variant). We report ROC-AUC (left) and PR-AUC (right). BSF operates on the intermediate denoiser representations, while ESM-2-650M scores the clean sequence prediction available at each timestep.

During generation, we record the per-residue representations from layer 9 of the denoiser, with dimensionality $d = 768$, at timesteps 9, 19, 30, 43, 58, 77, 100, 133, 185, and 341 for both variants. The BSF and AE are trained only on representations from the unlabeled training proteins. The final sequence for each protein is decoded from its final latent state. For an intermediate timestep t, the sequence used by the baselines and for the sequence-entropy analysis is decoded from the model's clean prediction at that timestep.

A.2.1 Labeling

We label generated protein sequences using two criteria namely high perplexity and low sequence entropy. These capture two different types of generation failures, where high perplexity identifies sequences that are unlikely under a pretrained protein language model, while low entropy identifies repetitive or low-complexity sequences.

High-perplexity sequences: To identify high-perplexity sequences, we use ProtT5-XL-UniRef50 (Elnaggar et al., 2020), a protein language model with approximately 3B parameters. ProtT5 follows an encoder-decoder architecture and was pretrained on the Big Fantastic Database (BFD) followed by training on UniRef50. The model uses the T5 span-corruption objective, where contiguous spans of the input sequence are replaced with sentinel tokens and the decoder is trained to reconstruct the missing residues. To obtain a pseudo-perplexity score for a generated sequence, we mask each residue position (or a small span) using a sentinel token in the encoder input and record the probability assigned by the decoder to the corresponding true residue. The negative log-probabilities are averaged across all positions in the sequence and exponentiated to obtain a sequence-level pseudo-perplexity score. Sequences with pseudo-perplexity above the selected threshold are labeled as high-perplexity failures.

Low-entropy sequences: We use sequence entropy to identify repetitive and low-complexity generations. For each generated sequence, we compute the Shannon entropy of its amino-acid composition as $-\sum_{a\in\mathcal{A}} p(a)\log_2 p(a)$, where $p(a)$ is the fraction of residues corresponding to amino acid a, and $\mathcal{A}$ denotes the 20 standard amino acids. The maximum possible entropy is $\log_2(20) \approx 4.32$ bits when all amino acids occur with equal frequency, while the entropy decreases as the sequence becomes dominated by a smaller number of amino acids. Low sequence entropy captures repetitive or low-complexity artifacts where the generative model repeatedly produces a limited set of residues or short motifs instead of a diverse protein sequence. We label sequences with entropy below the selected threshold as low-entropy failures. A generated sequence is considered a failed generation if it is labeled as either high-perplexity or low-entropy.

A.3 Baseline Implementation

We compare against three language-model baselines (two PLMs, and one LM), ESM-2 with 8M and 650M parameters Lin et al. (2022) and RoBERTa Liu et al. (2019). RoBERTa is a regular language model, which we use as a sequence-likelihood baseline without any protein specific training. All models use their released weights and are not fine-tuned.

For ESM-2, we use the corresponding amino-acid tokenizer. For RoBERTa, we separate residues by spaces such that each residue maps to a single token, and verify that the tokenized sequence decodes back to the original protein sequence. The detection score is the mean per-residue negative log-likelihood from a single unmasked forward pass. We compute the cross-entropy between the logits at each residue position and the corresponding residue, and average it across all residues while excluding the start and end tokens.

At an intermediate timestep t, the baselines score the sequence decoded from La-Proteina's clean prediction at that timestep, which is the most complete sequence available at that point in generation. Higher negative log-likelihood is treated as more indicative of generation failure. As with reconstruction-based markers, the score direction for each baseline is fixed on the validation set based on its validation ROC-AUC.

A.4 Training Details

All reconstruction models are trained only on intermediate representations from the unlabeled training generations. We train a separate model for each captured timestep using the per-residue layer-9 embeddings with dimensionality $d = 768$. For each timestep, the embeddings are standardized using statistics computed from the corresponding training embeddings. This gives 7.5M embeddings for LD1 (25,000 proteins $\times$ 300 residues) and 4.9M embeddings for LD2 (9,880 proteins $\times$ 500 residues). We hold out 5% of the train embeddings to monitor reconstruction quality. All models are trained with Adam for 40 epochs, and are retrained once with a longer schedule if the held-out reconstruction quality is poor ($R^2 < 0.7$) or if a large fraction of the dictionary remains unused.

A.4.1 BSF

We use the Grassmannian BSF Fel et al. (2026) with an expansion factor of 8, giving $G = 768$ blocks with block dimension $b = 8$. During training, each embedding is reconstructed using its top 16 ($K = 16$) blocks, while maintaining orthonormality of the blocks throughout training. Following Fel et al. (2026), we add Gaussian noise to the input during training with noise power equal to 10% of the signal power, which acts as a regularizer for the dictionary. At detection time, the number of retained blocks K is selected using the validation set, as described in Appendix A.5

A.4.2 AE

The AE is a fully connected autoencoder with architecture $768 \to 6144 \to 32 \to 6144 \to 768$ and GELU activations. It does not use any sparsity constraint and is trained without input noise. Since the AE does not have blocks, we compute its negative feature entropy using the magnitudes of the 32 bottleneck units.

A.5 Configuration Sweep on Validation Set-Ablations

For LD1 variant, we compare the reconstruction models at timestep 100 using the validation set. For each model, we vary the expansion factor and the number of retained features K, and compare the reconstruction residual, negative feature entropy, and their fusion. Table 4 reports the best K for each model and expansion factor.

We make three observations from this comparison. First, combining the two markers improves over either marker alone for every sparse model. For BSF with expansion $\times 8$, fusion improves the validation ROC-AUC from 0.746 using the residual and 0.704 using NFE to 0.805. Second, BSF performs better than the other reconstruction models at both expansion factors, while increasing the dictionary size does not improve performance. Third, detection performs best when only a small number of blocks are retained. For BSF with expansion $\times 8$, the fused ROC-AUC remains around

0.80 for K between 2 and 4, and decreases as K increases, reaching 0.756 at $K = 16$ and 0.634 at $K = 64$ (not shown in the table). Retaining more blocks improves reconstruction, but makes both the residual and block usage less specific to the eventual failure. The same pattern is observed on the test set for the selected BSF, where the residual alone gives a ROC-AUC of 0.749 compared to 0.786 with fusion.

Table 4: Validation ROC-AUC of the reconstruction residual (R), negative feature entropy (NFE), and their fusion on LD1 at timestep 100. For each model and expansion factor, we report the K with the best fused ROC-AUC. The AE is dense and has no K.

Model	Expansion	K	R	NFE	Fused ROC-AUC	Fused PR-AUC
BSF	$\times 8$	4	0.746	0.704	**0.805**	**0.248**
BSF	$\times 16$	4	0.749	0.678	0.793	0.218
Matryoshka SAE	$\times 8$	8	0.713	0.660	0.777	0.187
Matryoshka SAE	$\times 16$	12	0.695	0.663	0.771	0.164
BatchTopK SAE	$\times 8$	12	0.718	0.624	0.766	0.180
BatchTopK SAE	$\times 16$	8	0.739	0.607	0.760	0.184
AE	$\times 8$	–	0.630	0.613	0.701	0.113
AE	$\times 16$	–	0.632	0.715	0.703	0.118

A.6 Hyperparameters

Table 5 summarizes the hyperparameters used for the BSF and AE results reported in the main paper related to protein sequences.

Table 5: Hyperparameters of the BSF and AE used in the main results.

	BSF	AE
Representation	layer 9, $d = 768$	layer 9, $d = 768$
Architecture	$G = 768$ blocks, $b = 8$ (expansion $\times 8$)	768–6144–32–6144–768, GELU
Training sparsity	top-16 blocks	–
Detection sparsity K	4 (LD1), 6 (LD2)	–
Input noise	10% of signal power	–
Optimizer, learning rate	Adam, 10^{-3}	Adam, 3×10^{-4}
Batch size	8192 (LD1), 4096 (LD2)	4096
Epochs	40	40
Training embeddings	7.5M (LD1), 4.9M (LD2)	7.5M (LD1), 4.9M (LD2)
Detection timestep	100 (LD1), 43 (LD2)	100 (LD1), 43 (LD2)
Marker	fused	fused

A.7 Additional Reconstruction Models

In addition to BSF and AE, we evaluate two sparse autoencoders trained on the same embeddings using the same data and training schedule. The first is a BatchTopK SAE Bussmann et al. (2024); Gao et al. (2025), second is a Matryoshka SAE Bussmann et al. (2025); Zaigrajew et al. (2025), which additionally trains nested prefixes of its dictionary to reconstruct the input independently. Both models use a dictionary with $768 \times E$ latents for expansion factor $E \in \{8, 16\}$. Their negative feature entropy is computed from the magnitudes of the retained latents.

As shown in Table 4, both sparse autoencoders detect failures well above chance, with fused validation ROC-AUCs between 0.760 and 0.777. However, both remain below BSF at the two evaluated expansion factors. We therefore report BSF and AE in the main paper.

A.7.1 Additional Details on Failure Signal Analysis

Concept selection: For each sequence at timestep 100, we compute the share of activation assigned to every BSF concept. For a concept g, we sum its activation magnitude $\|z_g\|$ across residues where it is among the $K = 4$ active concepts, and divide by the total activation across all concepts. This gives the same concept distribution used for computing NFE. Distinctive concepts are selected using only the validation set. For each of the 768 concepts, we compare the mean activation share between failed and good validation sequences. Among the significant concepts with higher activation in failed sequences, we rank the concepts using their validation ROC-AUC and select the top

four. The selected LE concepts are 691, 38, 147, and 671, while the selected HP concepts are 200, 116, 577, and 98. None of these concepts is active on most residues or concentrated at the ends of the sequence.

Lead-time analysis: We compare a representation-based signal and an output-based signal at the ten timesteps (9, 19, 30, 43, 58, 77, 100, 133, 185, and 341 out of 400). The representation-based signal is the NFE score s_{NFE}, computed at each timestep using the BSF dictionary trained for that timestep and the same configuration used by the main detector. The output-based signal is the entropy of the amino-acid composition of La-Proteina's clean sequence prediction at that timestep. Thresholds are selected separately at each timestep using good validation sequences. We use the 95th percentile of NFE and the 5th percentile of sequence entropy (as no negative sign was used for sequence entropy), with the direction fixed in advance such that higher NFE and lower sequence entropy indicate failure. A test sequence is counted as flagged from the first timestep at which its signal crosses the corresponding threshold, so the reported percentages are cumulative. At timestep 100 for LD1, NFE flags 26% of low-entropy failures while sequence entropy flags 5%, compared to 8% and 7% of good sequences. At timestep 43 for LD2, NFE flags 68% of low-entropy failures while sequence entropy flags 3%, compared to 5% and 4% of good sequences.

A.8 Qualitative Examples

We provide qualitative examples comparing the residue distributions of good and low-entropy generations of protein sequences (Figure 8).

B Image Generation

B.1 Generation Setup

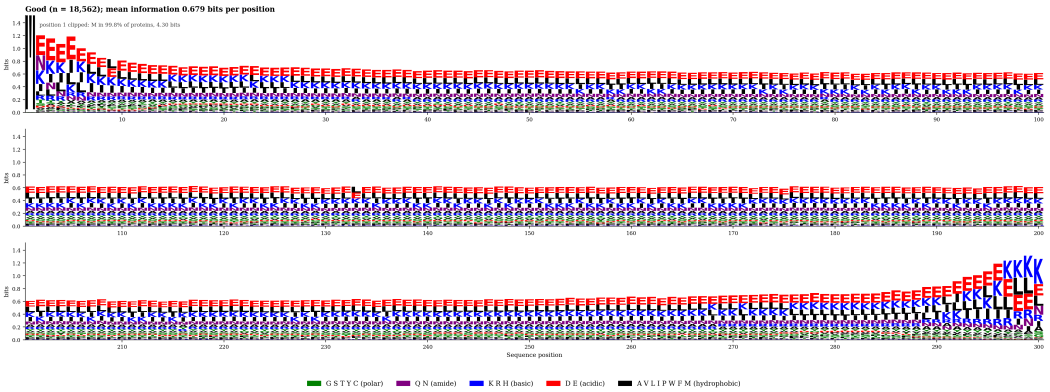
For AFHQv2, FFHQ, and 11kHands, we generate images using the 40-step EDM Euler sampler with the EDM noise schedule, where σ decreases from 80 to 0.002 with $\rho = 7$. For SimpleShapes and MNIST, we use 250-step ancestral DDPM sampling with the 1,000-step noise schedule of each model. The generated image resolutions are 64×64 RGB for AFHQv2 and FFHQ, 128×128 RGB for 11kHands, 28×28 grayscale for MNIST, and 64×64 grayscale for SimpleShapes.

For each dataset, we generate a separate set of 1,024 unlabeled images that is used only for training the BSF and AE, except for the MNIST AE, where we use 2,048 images. The labeled evaluation images are split into validation and test sets of 512/512 for AFHQv2, 511/513 for FFHQ, 256/256 for 11kHands and MNIST, and 490/490 for SimpleShapes.

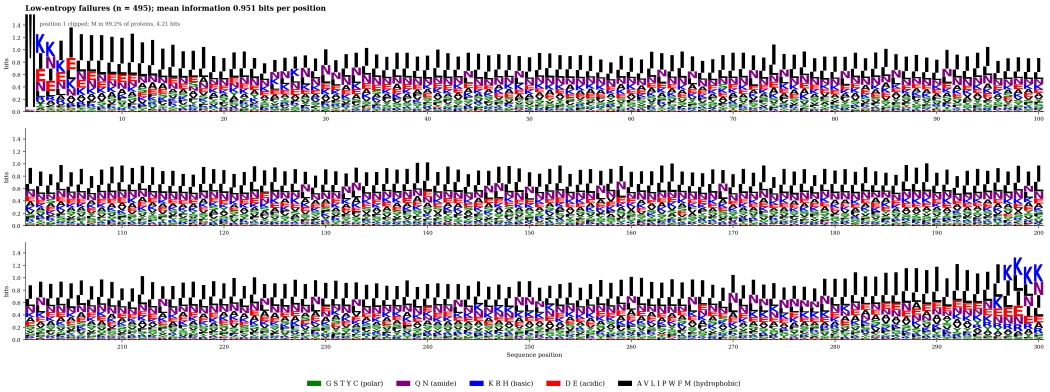
During sampling, we record the spatial activations from a UNet layer at a fixed set of sampling steps. The selected layer and detection step for each dataset are reported in Table 7. We report the detection point as the fraction of generation remaining, computed out of 40 sampler steps for EDM and out of 1,000 timesteps for DDPM.

B.1.1 Labeling

Image labeling. For AFHQv2, FFHQ, 11kHands, and MNIST, we manually label the generated images following the user study protocol of Sobieski et al. (2026). Two annotators independently label each dataset. Before labeling, each annotator goes through a calibration stage by viewing 128 real images from the corresponding training data. Following the prior work Sobieski et al. (2026), each generated image is labeled as a hallucination, correct, or ambiguous using a dataset-specific definition. For FFHQ and AFHQv2, hallucinations include misaligned, duplicated, missing, or fused facial features. For 11kHands, they include deformed, missing, extra, or merged fingers. For MNIST, an image is labeled as a hallucination if the generated digit is not recognizable as any digit from 0–9. Images are shown one at a time in a fixed shuffled order. The labeling process is blinded, such that annotators do not have access to the other annotator's labels or to any detector scores. Images labeled as ambiguous are excluded from evaluation. Since the annotators can disagree on subtle cases, we compute each evaluation metric separately using the labels from each annotator and report the average across the two. For SimpleShapes, failures are defined by the data-generating rules and are therefore labeled procedurally Aithal et al. (2024); Sobieski et al. (2026). An image is



(a) Good proteins ($n = 18{,}562$).



(b) Low-entropy failures ($n = 495$).

Figure 8: Sequence logos of the generated proteins (sequence length 300) from LD1. Letter height is residue frequency times the information content at that position, in bits. The proteins are unconditionally generated and unaligned, so each column compares residues at the same sequence index. The y-axis is capped above position 1; when position 1 exceeds the cap it is clipped and labeled with its residue and value. Colors - polar (green), amide (purple), basic (blue), acidic (red), hydrophobic (black).

considered a failure if it contains a shape in the wrong region, contains more shapes than allowed, has a malformed shape, or contains no shape.

B.2 Baseline Implementation

We implement LID Sobieski et al. (2026), LMI Sobieski et al. (2026), and TVF Aithal et al. (2024) on our generated images, models, and labels, following their original definitions. LID and LMI score the final generated image, while TVF uses the model's clean image predictions over a window of sampling steps and needs the generation only up to the end of that window.

For LID, we use the denoising score matching estimate averaged over 32 noise draws, and select the noise level on the validation set from 13 candidate values that include those reported in Sobieski et al. (2026). For LMI, we use 32 perturbations of the initial noise with the β values reported in Sobieski et al. (2026).

For TVF, we select the window on the validation set from 64 candidate windows spanning 2 to 16 consecutive sampler steps. For every baseline, both the configuration and score direction are selected using the mean validation ROC-AUC across both human annotators and are fixed before test evaluation.

B.3 Training Details

Each spatial position in a UNet feature map is treated as one embedding. All reconstruction models are trained on embeddings from 1,024 unlabeled generated images per dataset, except for the MNIST AE where we use 2,048 images. The embeddings are standardized using statistics computed from the corresponding training embeddings. We train a separate model for each layer, representation, and timestep, and hold out a small portion of the training embeddings to monitor reconstruction quality.

B.3.1 BSF

We use the Grassmannian BSF with block size 8. The models are trained with Adam using a learning rate of 3×10^{-3} and batch size 8,192. During training, we add Gaussian noise to the input with standard deviation equal to 0.1 times the batch standard deviation. Each model is trained for 40 epochs and retrained once for 80 epochs if the held-out reconstruction is poor ($R^2 < 0.7$) or if more than 30% of the blocks remain unused.

B.3.2 AE

The AE is a GELU MLP with a bottleneck of half the embedding dimension, using width 128 for all datasets except MNIST, where the bottleneck width is 16. We train the AE with Adam using a learning rate of 10^{-3}, and 3×10^{-4} for MNIST, with Gaussian input noise of standard deviation 0.1. The models are trained for up to 40 epochs with early stopping, and we keep the checkpoint with the best held-out reconstruction.

B.4 Configuration Sweep on Validation Set-Ablations

For AFHQv2, FFHQ, 11kHands, and MNIST, we perform a validation sweep over three UNet layers, two representations (the full layer output or only the update added by the layer), all saved timesteps, BSF expansion factors $\{8, 16, 32\}$ with training $K \in \{8, 16, 32, 64\}$, and six AE architectures. This results in a total of 6,674 trained models, each evaluated on the validation set.

Since LID and LMI only produce a score after the image has been fully generated, we are interested in how early reconstruction-based detectors can match their performance. We therefore select the earliest timestep at which a detector reaches the validation ROC-AUC of the strongest baseline. If no timestep reaches the baseline, we select the timestep with the best validation ROC-AUC. For the AE on AFHQv2 and SimpleShapes, whose best validation timesteps are at the end of generation, we instead select the best timestep with 20-50% of generation remaining. The layer, representation, dictionary size, K, and marker are all selected using the validation set. Table 6 reports the best validation ROC-AUC for each design choice within 15-50% of generation remaining. The reconstruction residual is the stronger single marker on all four datasets, while the best layer and representation vary across datasets.

Table 6: Best validation ROC-AUC for each design choice, over all other settings, with 15–50% of generation remaining.

		AFHQv2	FFHQ	11kHands	MNIST
Model	BSF	**0.673**	**0.778**	0.809	0.755
	AE	0.666	0.768	**0.814**	**0.838**
Layer	mid	0.669	0.759	**0.814**	**0.838**
	up11	0.664	0.775	0.752	0.775
	up12	**0.673**	**0.778**	0.741	0.728
Representation	layer output	0.669	**0.778**	0.809	**0.838**
	layer update	**0.673**	0.759	**0.814**	0.827
Marker	residual	**0.673**	**0.778**	**0.814**	**0.838**
	NFE	0.669	0.759	0.745	0.741

B.5 Hyperparameters

Table 7 summarizes the configurations used for the main image results.

Table 7: Configurations used in the main image results. Remaining is the fraction of generation still remaining at the detection point. E is expansion factor; K_{train} / K is top-K blocks during training / at detection.

	BSF					AE		
Dataset	Layer	Steps left	Marker	E	K_{train} / K	Layer	Steps left	Widths
AFHQv2	up12	45%	fused	16	8 / 4	up12	22.5%	256–1024–128
FFHQ	up11	45%	residual	16	8 / 8	up11	45%	256–1024–128
11kHands	mid	37.5%	residual	32	8 / 4	mid (update)	30%	256–1024–128
MNIST	up11	28.4%	fused	32	32 / 32	mid	28.4%	32–1024–1024–16
SimpleShapes	down2	15.2%	residual	8	16 / 2	down2	28.4%	256–1024–128

B.6 Additional Reconstruction Models

Table 8: Best validation ROC-AUC of each reconstruction model.

Model	AFHQv2	FFHQ	11kHands	MNIST	SimpleShapes
BSF	0.635	**0.756**	**0.779**	0.695	**0.853**
AE	**0.659**	0.744	0.754	**0.805**	0.486
TopK SAE	0.583	0.549	0.708	0.587	0.659
Matryoshka SAE	0.562	0.703	0.633	0.540	–

We also evaluate a TopK SAEs with an expansion factor 8 and training $K \in \{64, 128, 256, 512\}$ and expansion factors 16 and 32 with $K = 128$, and Matryoshka SAE with expansion factor 8 and $K = 128$. These models are trained on the same embeddings and are evaluated in an validation search using one layer per dataset, where BSF and AE are also included for comparison. Table 8 compares the best validation ROC-AUC of each model using either the reconstruction residual or NFE marker within 15-50% of generation remaining. BSF outperforms both SAE variants on every dataset, and the AE outperforms them on every dataset except SimpleShapes, so we report BSF and AE in the main paper.