RESIDUALIZED TEMPORAL SPARSE AUTOENCODERS FOR INTERPRETING DIFFUSION MODELS

Calvin Yeung*, **Prathyush Poduval, Ali Zakeri, Zhuowen Zou, Mohsen Imani**
Department of Computer Science
University of California, Irvine
Irvine, CA 92697
*Corresponding author: chyeung2@uci.edu

May 28, 2026

ABSTRACT

Text-to-image diffusion models generate images through an iterative denoising process, so internal neural layers produce trajectories of activations rather than single static representations. Sparse autoencoders (SAEs) have recently been used to decompose diffusion activations into interpretable feature directions, but most approaches analyze activations at individual timesteps or condition on time rather than learning directly from full activation trajectories. In this work, we introduce residualized temporal SAEs for diffusion activation trajectories. We collect activations across denoising time, fit linear predictors between neighboring timesteps, and represent each trajectory using an initial activation together with residual components not explained by these linear dynamics. Training an SAE on this residualized representation encourages sparse latents to capture structure beyond what is linearly predictable. The residualized decoder directions can be mapped back into activation space, allowing each latent to be analyzed as a feature trajectory over denoising time. Through reconstruction and ablation studies, spatiotemporal feature analysis, and qualitative steering experiments on Stable Diffusion 1.5, we show that residualized temporal SAEs provide a useful framework for studying temporally structured diffusion activations.

1 Introduction

Text-to-image diffusion models generate images through an iterative denoising process, repeatedly applying the same denoising network across many timesteps (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021). As a result, an internal layer of the model does not produce a single static representation for an image, but rather a trajectory of representations that evolves over the course of generation. Early denoising steps operate on noisy latents and may encode coarse semantic or compositional structure, while later steps refine spatial details and visual appearance. This timestep-dependent view is consistent with analyses showing that denoising at particular noise levels provides a useful pretext task for learning rich visual concepts (Choi et al., 2022), and with work showing that intermediate diffusion activations can serve as semantic visual representations (Baranchuk et al., 2022; Zhang et al., 2023). Understanding these activation trajectories is therefore important for interpreting how diffusion models organize and transform visual information over time.

Sparse autoencoders (SAEs) have emerged as a useful tool for decomposing neural network activations into sparse, interpretable feature directions (Bricken et al., 2023; Gao et al., 2025). Recent work has begun to apply SAEs to text-to-image diffusion models, showing that sparse features in diffusion activations can be interpretable and can causally influence generation (Surkov et al., 2025; Tinaz et al., 2025; Shabalin et al., 2025). Tinaz et al. (2025) analyze how SAE concepts emerge and evolve through denoising time, while Huang et al. (2026) propose temporal-aware SAEs for Diffusion Transformers. Other work uses SAE features for diffusion-model concept editing or unlearning (Cywiński and Deja, 2025; Kim and Ghadiyaram, 2025). Together, these works show that diffusion activations contain interpretable sparse structure.

However, existing approaches leave open how to learn sparse features from activation trajectories. This differs from learning features from isolated timestep activations (Surkov et al., 2025; Shabalin et al., 2025), timestep-conditioned representations (Tinaz et al., 2025), or architectures such as Diffusion Transformers (Huang et al., 2026). A natural way to incorporate temporal information is to train an SAE on concatenated activation trajectories. However, diffusion activations are highly correlated across adjacent timesteps. Much of the activation at one timestep is linearly predictable from the activation at a neighboring timestep. A temporal SAE trained directly on raw activations may therefore spend capacity modeling linearly predictable, redundant information. This suggests that temporal SAE models should account for both the sequential nature of diffusion activations and their strong temporal correlation.

Motivated by this observation, we introduce a residualized temporal SAE for temporally structured diffusion activations. Rather than training directly on raw activation trajectories, we separate each trajectory into a predictable component and a residual component, then train an SAE on the resulting residualized representation. This encourages sparse features to capture structure beyond linearly predictable information while still preserving their interpretation as feature trajectories over denoising time. The learned latents can be interpreted as feature trajectories: by mapping residualized decoder directions back into activation space, we obtain a sequence of directions for each latent. This lets us measure when a feature is expressed, where it localizes spatially, and whether it has visible generative effects when intervening during generation.

In this work, (1) we introduce residualized temporal SAEs for diffusion activation trajectories, in which sparse latents are encouraged to capture variation beyond linearly predictable dynamics through residualization; (2) we derive an activation-space interpretation of residualized decoder directions, allowing each latent to be analyzed as a feature trajectory with spatiotemporal structure; and (3) we empirically characterize these representations on Stable Diffusion 1.5 through reconstruction and ablation studies, spatiotemporal feature analysis, and qualitative steering, showing that the learned latents capture temporally structured, spatially localized directions with semantically meaningful generative effects.

2 Background

2.1 Text-Conditioned Latent Diffusion Models

Diffusion models generate data by learning to reverse a gradual noising process (Sohl-Dickstein et al., 2015; Ho et al., 2020). Text-to-image latent diffusion models, such as Stable Diffusion, perform this process in a learned latent space: a U-Net denoiser iteratively removes noise from a latent variable while conditioning on a text prompt (Rombach et al., 2022). We denote the denoising network by $\epsilon_\theta(\mathbf{x}_\tau, \tau, \mathbf{c})$, where $\mathbf{x}_\tau$ is the noisy latent at denoising step τ and $\mathbf{c}$ is an optional conditioning signal.

Text conditioning is injected into the U-Net through cross-attention layers. Text-to-image models commonly use classifier-free guidance (CFG), in which the U-Net is evaluated with both the text condition $\mathbf{c}$ and an unconditional condition $\varnothing$ (Ho and Salimans, 2022):

$$\epsilon_{\text{cfg}}(\mathbf{x}_\tau, \tau, \mathbf{c}) = \epsilon_\theta(\mathbf{x}_\tau, \tau, \varnothing) + w\left(\epsilon_\theta(\mathbf{x}_\tau, \tau, \mathbf{c}) - \epsilon_\theta(\mathbf{x}_\tau, \tau, \varnothing)\right), \tag{1}$$

where w is the guidance scale. Since prompt information enters through the conditional branch and its cross-attention layers, we focus on conditional U-Net activations from a cross-attention block.

Internal diffusion activations have been used as representations for segmentation, attribution, correspondence, and control. Prior work has shown that DDPM and Stable Diffusion activations provide useful pixel-level and semantic features (Baranchuk et al., 2022; Tang et al., 2023a; Zhang et al., 2023), while cross-attention maps support word–pixel attribution (Tang et al., 2023b). Other methods consolidate diffusion features across layers and timesteps to obtain stronger descriptors (Luo et al., 2023). These results motivate studying diffusion activations as rich representations. For a fixed internal U-Net layer, repeated denoising produces an activation trajectory $\{\mathbf{a}(\tau)\}$ over denoising steps. This temporal structure is central to our method: rather than analyzing activations independently at each timestep, we study sparse features that evolve across activation trajectories.

2.2 Sparse Autoencoders

Sparse autoencoders (SAEs) decompose neural network activations into sparse combinations of learned feature directions (Bricken et al., 2023). Given an activation vector $\mathbf{x} \in \mathbb{R}^d$, an SAE maps it to a sparse latent representation $\mathbf{h} \in \mathbb{R}^m$ and reconstructs the input:

$$\mathbf{u} = W^{\text{enc}}\mathbf{x} + \mathbf{b}^{\text{enc}}, \qquad \mathbf{h} = S(\mathbf{u}), \qquad \mathbf{x}^{\text{rec}} = W^{\text{dec}}\mathbf{h} + \mathbf{b}^{\text{dec}}. \tag{2}$$

where S is a sparsifying nonlinearity or operator. The columns of W^{dec} are interpreted as learned feature directions. SAEs are trained to reconstruct activations while keeping the latent code sparse:

$$\mathcal{L}_{\mathrm{SAE}}(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^{\mathrm{rec}}\|_2^2, \tag{3}$$

together with a sparsity penalty or constraint on $\mathbf{h}$. Common choices include an ℓ_1 penalty, Top-K sparsity, and BatchTopK sparsity (Gao et al., 2025; Bussmann et al., 2024). In this work, we use BatchTopK SAEs, which enforce sparsity by retaining only the largest latent activations across a minibatch.

In diffusion models, SAEs can be trained on U-Net or transformer activations to identify features associated with visual or semantic concepts (Surkov et al., 2025; Tinaz et al., 2025; Shabalin et al., 2025). This motivates using SAEs to study not only individual activations, but also activation trajectories across denoising time.

3 Methods

3.1 Collecting Activation Trajectories

Consider a diffusion model that generates a sample $\mathbf{x}_0$ from Gaussian noise through a sequence of denoising steps. We use τ to denote a full denoising step in the sampler and extract an activation tensor $\mathbf{a}(\tau)$ from a fixed layer of the denoising network ϵ_θ. For example, in a U-Net denoiser, $\mathbf{a}(\tau)$ may correspond to a bottleneck activation (Ronneberger et al., 2015). The full activation trajectory is the sequence of these layer activations over the denoising process.

In this work, we study text-to-image diffusion models, specifically Stable Diffusion 1.5. Because text conditioning has been shown to produce interpretable SAE features (Tinaz et al., 2025), we collect activations from the conditional branch of the U-Net during classifier-free guidance. In particular, we extract the residual contribution of the mid-block cross-attention layer, defined as the layer output minus its input. This isolates the contribution added by the residual connection of that block.

Each collected activation has shape $H \times W \times d$, where H and W are spatial dimensions and d is the channel dimension. To simplify notation, we let q index one spatial token from one generated sample. We subsample the denoising trajectory every ℓ sampler steps and use $i = 0, \ldots, T-1$ for the resulting subsampled timestep indices. If τ_i is the full sampler step associated with subsampled index i, then $\mathbf{a}_{q,i} \in \mathbb{R}^d$ denotes the activation for token trajectory q at full step τ_i. Further details are included in Appendix A.

3.2 Linear Residualization

Activation trajectories in diffusion models are highly correlated across time. We first normalize activations separately at each subsampled timestep using training-set statistics, $\bar{\mathbf{a}}_{q,i} = (\mathbf{a}_{q,i} - \boldsymbol{\mu}_i^a) \odot (\boldsymbol{\sigma}_i^a)^{-1}$, where $\boldsymbol{\mu}_i^a$ and $\boldsymbol{\sigma}_i^a$ are the per-coordinate mean and standard deviation for activation block i.

We then remove the linearly predictable component between consecutive normalized activation blocks. For each adjacent subsampled transition $i-1 \to i$, $i = 1, \ldots, T-1$, we fit a ridge regressor $(W_i, \mathbf{b}_i)$ shared across all training samples and spatial positions:

$$\min_{W_i, \mathbf{b}_i} \sum_q \|\bar{\mathbf{a}}_{q,i} - (W_i \bar{\mathbf{a}}_{q,i-1} + \mathbf{b}_i)\|_2^2 + \lambda \|W_i\|_F^2. \tag{4}$$

As is standard in ridge regression, the bias is unregularized. The residual is $\mathbf{r}_{q,i} = \bar{\mathbf{a}}_{q,i} - (W_i \bar{\mathbf{a}}_{q,i-1} + \mathbf{b}_i)$ for $i = 1, \ldots, T-1$. Thus, $\mathbf{r}_{q,i}$ captures the component of the normalized activation at timestep i that is not linearly predictable from the previous normalized activation in the subsampled trajectory.

3.3 Temporal Sparse Autoencoder

For each token trajectory q, we construct a residualized trajectory vector

$$\mathbf{z}_q = [\bar{\mathbf{a}}_{q,0}, \mathbf{r}_{q,1}, \ldots, \mathbf{r}_{q,T-1}] \in \mathbb{R}^{Td}. \tag{5}$$

The first term anchors the trajectory, while the remaining terms describe deviations from the fitted linear dynamics.

Before SAE training, we apply a second normalization to the concatenated representation. This normalization is performed separately for each concatenated component. Let $\mathbf{z}_{q,0} = \bar{\mathbf{a}}_{q,0}$ and $\mathbf{z}_{q,i} = \mathbf{r}_{q,i}$ for $i \geq 1$. For each component, we define $\mathbf{x}_{q,i} = (\mathbf{z}_{q,i} - \boldsymbol{\mu}_i^z) \odot (\boldsymbol{\sigma}_i^z)^{-1}$, where $\boldsymbol{\mu}_i^z$ and $\boldsymbol{\sigma}_i^z$ are computed from the training split. The normalized SAE input is $\mathbf{x}_q = [\mathbf{x}_{q,0}, \mathbf{x}_{q,1}, \ldots, \mathbf{x}_{q,T-1}]$. Equivalently, we write $\mathbf{x}_q = \mathrm{norm}_z(\mathbf{z}_q)$ and $\mathbf{z}_q^{\mathrm{rec}} = \mathrm{unnorm}_z(\mathbf{x}_q^{\mathrm{rec}})$.

The SAE encodes and reconstructs the normalized residualized trajectory as $\mathbf{h}_q = f_{\text{enc}}(\mathbf{x}_q)$ and $\mathbf{x}_q^{\text{rec}} = f_{\text{dec}}(\mathbf{h}_q)$. We use BatchTopK SAEs, described in Section 2.2, with average sparsity K_{avg}. We train with the reconstruction loss

$$\mathcal{L}_{\text{SAE}} = \sum_{q \in \mathcal{B}} \left\| \mathbf{x}_q - \mathbf{x}_q^{\text{rec}} \right\|_2^2 + \beta \mathcal{L}_{\text{aux}}, \tag{6}$$

where $\mathcal{L}_{\text{aux}}$ is the auxiliary loss for dead-latent recovery as in Bussmann et al. (2024). Refer to Appendix A for specific implementation details.

3.4 Model Variants

We compare variants along two axes: whether the input is residualized, and whether temporal blocks are concatenated into a single trajectory-level SAE input:

$$\mathbf{z}_q^{\text{resid}} = [\bar{\mathbf{a}}_{q,0}, \mathbf{r}_{q,1}, \ldots, \mathbf{r}_{q,T-1}], \qquad \mathbf{z}_q^{\text{raw}} = [\bar{\mathbf{a}}_{q,0}, \bar{\mathbf{a}}_{q,1}, \ldots, \bar{\mathbf{a}}_{q,T-1}]. \tag{7}$$

Before SAE training, each concatenated component is normalized using its own training-set statistics, as in Section 3.2. In the concatenated variants, a single SAE is trained on the full trajectory vector. In the non-concatenated variants, we train a separate SAE for each subsampled timestep block.

For reconstruction comparisons, we distinguish two sparsity-matching protocols. In the main text, we match the total trajectory-level sparse-code budget: concatenated models use K_{avg} active latents on average for the full trajectory, and timestep-wise models distribute the same total average budget across the T independently reconstructed timestep blocks. This gives a direct comparison between trajectory-level and timestep-wise representations under a fixed number of active latents per token trajectory. In Appendix B, we also report a per-timestep-matched comparison, where each timestep-wise SAE uses K_{avg} active latents per subsampled timestep block. That setting gives timestep-wise models an effective trajectory-level budget of approximately TK_{avg} and should therefore be interpreted as a reconstruction-favorable baseline for independent timestep-wise SAEs.

We additionally compare against a Matryoshka SAE baseline inspired by Matryoshka Representation Learning (Kusupati et al., 2022) and Matryoshka SAEs (Bussmann et al., 2025). This baseline is trained on non-residualized concatenated activation trajectories, with latent groups aligned to subsampled timestep blocks. Implementation details are provided in Appendix A.3.

3.5 Activation-Space Decoder Directions

For residualized temporal SAEs, decoder directions initially live in normalized SAE input space. Let $\mathbf{d}_k \in \mathbb{R}^{Td}$ denote the decoder vector for latent k. We first undo the SAE-input normalization by applying the component-wise scales, giving $\mathbf{d}_k^z = \boldsymbol{\sigma}^z \odot \mathbf{d}_k = [\mathbf{v}_{k,0}, \mathbf{u}_{k,1}, \ldots, \mathbf{u}_{k,T-1}]$, where $\mathbf{v}_{k,0}$ is the direction for the first normalized activation block and $\mathbf{u}_{k,i}$ is the direction for residual block i.

We then map this residualized direction back through the fitted linear dynamics to obtain a decoder trajectory in normalized activation space:

$$\bar{\boldsymbol{\phi}}_{k,0} = \mathbf{v}_{k,0}, \qquad \bar{\boldsymbol{\phi}}_{k,i} = W_i \bar{\boldsymbol{\phi}}_{k,i-1} + \mathbf{u}_{k,i}, \qquad i = 1, \ldots, T-1. \tag{8}$$

The bias terms $\mathbf{b}_i$ do not appear because decoder vectors represent directions rather than absolute activations. Finally, we undo the activation normalization at each timestep, $\boldsymbol{\phi}_{k,i} = \boldsymbol{\sigma}_i^a \odot \bar{\boldsymbol{\phi}}_{k,i}$, to obtain activation-space decoder trajectories in the original activation units $\{\boldsymbol{\phi}_{k,i}\}_{i=0}^{T-1}$, which we use for temporal analysis, spatial localization, and steering.

4 Results

4.1 Experimental Setup

Following prior work on sparse autoencoders for diffusion models (Surkov et al., 2025; Tinaz et al., 2025), we generate 50,000 training images and 2,500 validation images from LAION-COCO-aesthetic captions (Schuhmann et al., 2022) using Stable Diffusion 1.5. We collect U-Net activation trajectories according to the procedure described in Section 3.1 and evaluate the model variants defined in Section 3.4. All quantitative metrics are computed on the held-out validation set and are evaluated in the original activation space rather than in the normalized SAE input space. Additional implementation details are provided in Appendix A.

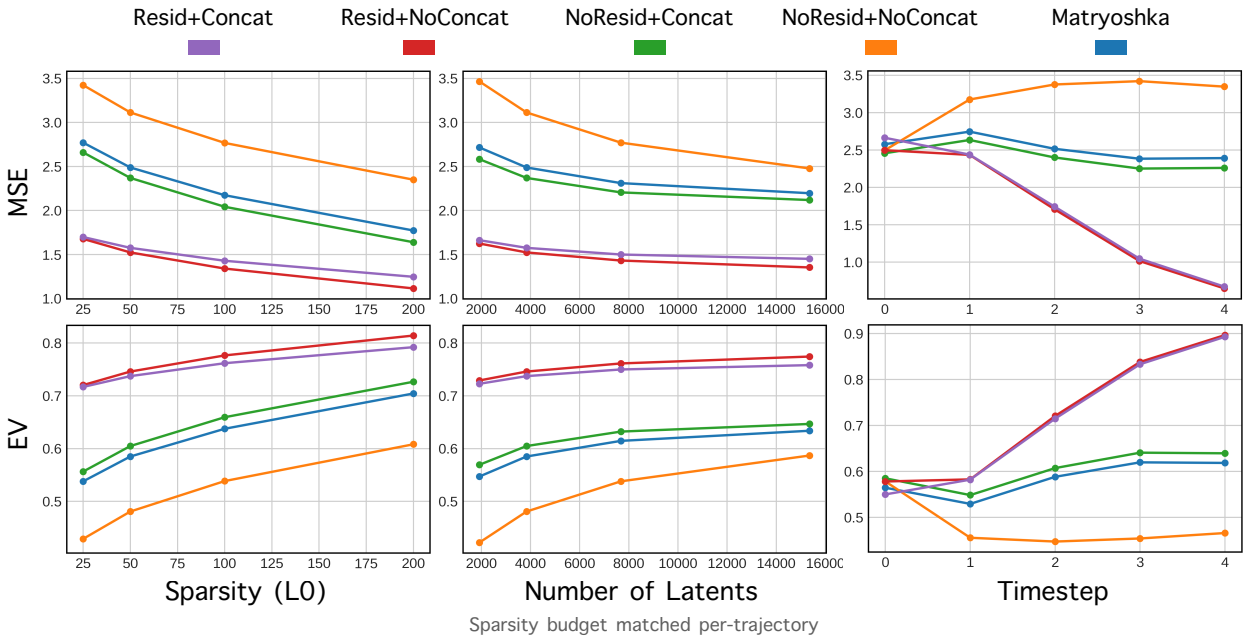

Figure 1: Activation-space reconstruction for BatchTopK SAE variants under a trajectory-level sparsity-budget match. Overall MSE/EV are shown versus total trajectory-level sparsity and dictionary size; per-timestep curves use the representative configuration $K_{\text{avg}} = 50$ and expansion factor 0.5. Timestep-wise baselines are assigned the same total average number of active latents per token trajectory as concatenated models. Residualized variants outperform their non-residualized counterparts, and the residualized concatenated trajectory SAE remains competitive while using a single trajectory-level sparse code. The Matryoshka curve is a non-residualized, component-normalized Matryoshka baseline.

4.2 Reconstruction Performance

We evaluate teacher-forced activation-space reconstruction of the original subsampled activation trajectory $\{\mathbf{a}_{q,i}\}_{i=0}^{T-1}$ for each spatial-token trajectory q. Metrics are computed in the original activation space rather than the normalized SAE input space. For residualized models, reconstructed residual blocks are mapped back through the fitted linear predictors and activation normalizers; for non-residualized models, reconstructed blocks are directly unnormalized. Non-concatenated models are reconstructed independently per timestep, and metrics are aggregated across subsampled timesteps. We report MSE, EV, and per-timestep EV, with full reconstruction details and metric definitions in Appendix A.4.

Effect of model variant, sparsity, and number of latents. Unless otherwise varied, reconstruction experiments use subsampling stride $\ell = 10$, trajectory-level BatchTopK average sparsity $K_{\text{avg}} = 50$, and expansion factor 0.5. We compare the four combinations of residualization and temporal concatenation, along with the non-residualized, component-normalized Matryoshka SAE baseline. The main comparison matches the total average number of active latents per token trajectory, so timestep-wise baselines do not receive a larger trajectory-level sparse-code budget than concatenated models.

Figure 1 summarizes the reconstruction results. At matched trajectory-level sparsity, residualized variants outperform their non-residualized counterparts. The direct comparison between Resid+Concat and NoResid+Concat shows that residualization improves reconstruction over raw activation concatenation across sparsity and dictionary-size settings. Resid+Concat is also competitive with the residualized timestep-wise baseline while using a single sparse code for the full denoising trajectory, which is the representation used for temporal-profile analysis, spatial localization, and steering. Per-timestep curves show that residualized variants improve most strongly at later subsampled timesteps. Appendix B reports the per-timestep-matched setting, where timestep-wise baselines receive a larger effective trajectory-level sparse-code budget and therefore serve as a reconstruction-favorable comparison.

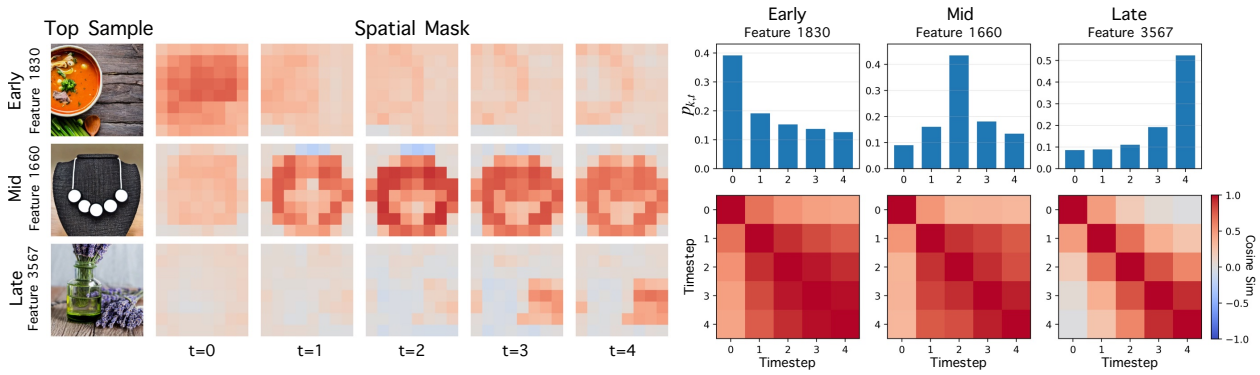

Figure 2: Representative early, middle, and late temporal SAE latents. Left: spatial cosine-similarity maps between activation-space decoder directions and U-Net tokens. Right: temporal profiles and cross-timestep decoder self-similarity.

4.3 Spatiotemporal Structure of Learned Latents

We next analyze the spatiotemporal structure of latents learned by concatenated trajectory SAEs. Unless otherwise specified, this analysis uses the residualized concatenated trajectory SAE with subsampling stride $\ell = 10$, $K_{\text{avg}} = 64$, and expansion factor 0.5. Each latent has an activation-space decoder trajectory $\{\boldsymbol{\phi}_{k,i}\}_{i=0}^{T-1}$. This allows us to study both when a latent is active over denoising time and where it localizes spatially during generation.

Temporal profiles. To summarize when each latent is most strongly expressed, we define its temporal profile as the normalized activation-space decoder magnitude $p_{k,i} = \left\|\boldsymbol{\phi}_{k,i}\right\|_2 / \sum_{j=0}^{T-1} \left\|\boldsymbol{\phi}_{k,j}\right\|_2$, which measures which subsampled timesteps are most strongly associated with latent k. We also compute pairwise cosine similarities between $\boldsymbol{\phi}_{k,i}$ across subsampled timesteps to measure how stable the latent's activation-space direction is over denoising time. In Figure 2, the left panel shows spatial cosine-similarity maps for representative early, middle, and late latents, while the right panel shows each latent's temporal profile and cross-timestep self-similarity. These examples illustrate that some latents have broad early support, whereas others peak in the middle or late parts of the denoising process.

To summarize temporal specialization across latents, we divide the denoising trajectory into early, middle, and late segments by checking whether the majority of the temporal mass is in the early, middle, or late third of the sequence. Early subsampled timesteps refer to the beginning of the reverse denoising process, when the latent image is still relatively noisy, and late indices refer to the end of denoising. Appendix D and Figure 8 further quantify this temporal structure by measuring cross-timestep decoder-direction self-similarity: early features tend to have higher off-diagonal self-similarity than middle and late features, suggesting that early features are more temporally persistent while later features are more timestep-specific.

Spatial localization. For a latent k, we compute the cosine similarity between its activation-space decoder vector $\boldsymbol{\phi}_{k,i}$ and each spatial activation token $\mathbf{a}_{n,i}^{(u,v)}$ for a given activation trajectory. This produces an $H \times W$ similarity map at each subsampled timestep index, indicating which spatial regions align most strongly with the latent over the denoising trajectory. Figure 2 visualizes these spatial maps over time for representative early, middle, and late latents, with each row corresponding to one selected latent. Appendix D and Figure 9 report positive spatial-mask entropy across temporal groups and timesteps. Averaging the mean normalized entropy over subsampled timesteps gives 0.876 for early features, 0.835 for middle features, and 0.772 for late features, indicating that early features have broader positive spatial support while middle and late features are more spatially concentrated.

4.4 Steering with Temporal SAE Features

The temporal SAE is trained on a subsampled denoising trajectory, using every ℓ-th denoising step, while steering is applied during the full denoising process. To apply SAE feature trajectories at all denoising steps, we expand each subsampled-index steering trajectory piecewise-constantly: every full denoising step between two subsampled timesteps uses the same steering vector as the corresponding subsampled timestep block.

All steering interventions are applied to the conditional branch of the classifier-free guidance U-Net evaluation; the unconditional branch is left unchanged. For the single-feature steering experiments below, we use the same residualized

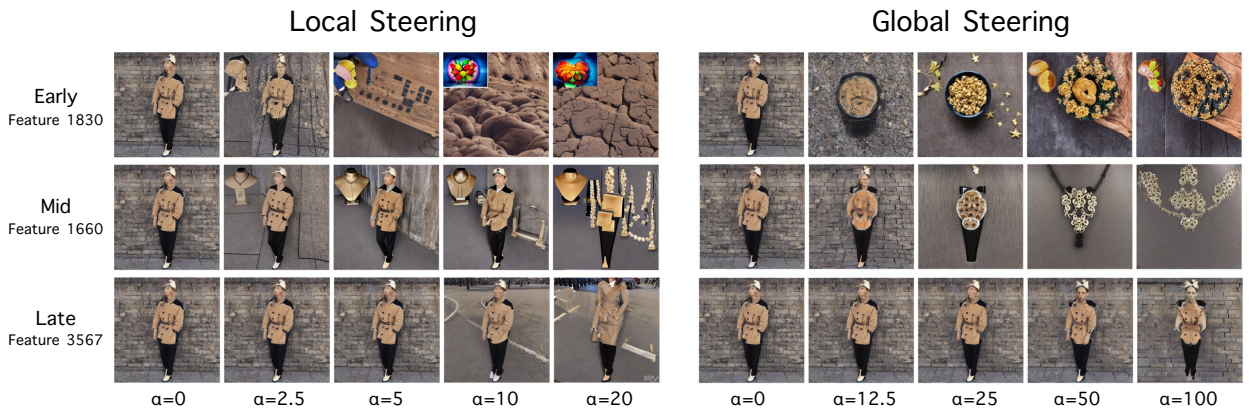

Figure 3: Single-feature steering with representative early, middle, and late latents. Local steering intervenes on one spatial token; global steering applies the same activation-space feature trajectory to all tokens.

concatenated trajectory SAE analyzed above, with subsampling stride $\ell = 10$, $K_{\text{avg}} = 64$, and expansion factor 0.5. In addition, we use empty-prompt generation and a guidance scale of 7.5. This setting isolates the visual effect of individual SAE feature trajectories without confounding from prompt-specific semantics. For the feature-transfer steering experiments below, we use temporal SAEs with subsampling stride $\ell = 10$, $K_{\text{avg}} = 50$, and expansion factor 0.5. Both steering settings use the same intervention rule. Let $\tilde{\mathbf{v}}_\tau$ denote the expanded activation-space steering direction at full denoising step τ, and let $\mathcal{S}_\tau$ denote the set of intervened spatial tokens. We update

$$\mathbf{a}_\tau^{(u,v)} \leftarrow \mathbf{a}_\tau^{(u,v)} + \alpha \left\| \mathbf{a}_\tau^{(u,v)} \right\|_2 \tilde{\mathbf{v}}_\tau, \qquad (u,v) \in \mathcal{S}_\tau, \tag{9}$$

and leave all tokens outside $\mathcal{S}_\tau$ unchanged. For global steering, $\mathcal{S}_\tau$ is the full spatial grid. For local steering, $\mathcal{S}_\tau$ contains only the chosen spatial token or region. The coefficient α specifies the steering strength.

Single-feature steering. We first evaluate whether individual temporal SAE features have visible generative effects. This experiment is intended as a qualitative test of the learned feature trajectories rather than as a benchmark for image-editing performance. We select representative early, middle, and late latents according to their temporal profiles and intervene using their activation-space decoder trajectories $\{\boldsymbol{\phi}_{k,i}\}_{i=0}^{T-1}$. For a selected latent k, we set the subsampled steering direction to $\mathbf{v}_i = \boldsymbol{\phi}_{k,i}$.

Figure 3 shows qualitative examples of local and global steering for these representative latents. Local interventions primarily affect a restricted image region, such as the upper-left corner, while global interventions alter the overall image appearance. The early and middle features produce interpretable changes, while the late feature does not produce a clear semantic effect in this example. Local steering with the middle feature is more spatially contained than local steering with the early feature: the early intervention produces stronger changes outside the selected region. This is consistent with the spatial maps in Figure 2, where the early feature has more diffuse spatial support, whereas the middle feature is more localized.

We also observe qualitative consistency between local and global interventions. For example, the middle feature adds a torso with jewelry under local steering, while global steering shifts the whole image toward jewelry-like visual structure. This interpretation is supported by the highest-activating samples for the same feature in Figure 7 of Appendix D.1, which are also jewelry-related. We include additional steering examples in Appendix E.

Feature transfer steering. We evaluate source-to-target feature transfer using paired prompts from the RIEBench dataset (Surkov et al., 2025). For each prompt pair, we run the diffusion model under both the source and target prompts and collect the corresponding activation trajectories. Each trajectory is encoded with the temporal SAE, and a subset of features $\mathcal{K}$ is selected for transfer, as described in Appendix E.2. During source-conditioned generation, we steer the source trajectory toward the target trajectory by adding the target decoded feature directions and subtracting the corresponding source decoded feature directions. Figure 4 showcases a few examples of the feature transfer steering.

Let $\{\boldsymbol{\phi}_{k,i}^{\text{src}}\}_{i=0}^{T-1}$ and $\{\boldsymbol{\phi}_{k,i}^{\text{tgt}}\}_{i=0}^{T-1}$ denote the activation-space decoded trajectories for feature k under the source and target prompts, respectively. For non-residual models, these activation-space trajectories are simply the decoder weights, as no reconstruction is needed. For each selected feature $k \in \mathcal{K}$, we define the source-to-target transfer direction

$$\Delta\boldsymbol{\phi}_{k,i} = \lambda^{\text{tgt}} \boldsymbol{\phi}_{k,i}^{\text{tgt}} - \lambda^{\text{src}} \boldsymbol{\phi}_{k,i}^{\text{src}}, \qquad i = 0, \ldots, T-1. \tag{10}$$

Figure 4: Source-to-target SAE feature transfer. For each row, the source and target images are unsteered generations from the corresponding prompts. The steered image is generated from the source prompt, while selected SAE features from the target trajectory are transferred into the source denoising trajectory. The Resid+Concat model is used to generate the examples.

Table 1: RIEBench (Surkov et al., 2025) source-to-target steering evaluation averaged over prompt pairs. Grounded-SAM2 masks are generated from the RIEBench original and edited grounding prompts. Edit efficiency is measured as ΔCLIP-T per unit LPIPS-S.

Model	CLIP-T ↑	ΔCLIP-T ↑	LPIPS-S ↓	Eff. ↑
Source Baseline	29.423 ± 0.220	0.000 ± 0.000	0.000 ± 0.000	–
Target Baseline	32.526 ± 0.177	3.103 ± 0.201	0.571 ± 0.010	–
Resid+Concat	29.557 ± 0.230	0.134 ± 0.080	0.194 ± 0.009	**0.691**
Resid+NoConcat	$\mathbf{29.567 \pm 0.226}$	$\mathbf{0.144 \pm 0.118}$	0.321 ± 0.011	0.449
NoResid+Concat	29.507 ± 0.225	0.084 ± 0.082	$\mathbf{0.188 \pm 0.009}$	0.447
NoResid+NoConcat	29.491 ± 0.216	0.068 ± 0.139	0.391 ± 0.011	0.174
Matryoshka	29.463 ± 0.226	0.039 ± 0.072	0.189 ± 0.009	0.206

The per-timestep steering direction is then $\mathbf{v}_i = \sum_{k \in \mathcal{K}} \Delta\phi_{k,i}$. Thus, the image is still generated under the source prompt, but the selected SAE feature trajectories are shifted from their source-prompt directions toward their target-prompt directions. For localized interventions, the spatial intervention region $\mathcal{S}_\tau$ is selected using Grounded-SAM2 (Ren et al., 2024).

We evaluate feature transfer using CLIP similarity to the target prompt, the change in target-prompt CLIP similarity relative to the source baseline, and LPIPS distance to the source image. These metrics capture the tradeoff between importing target-associated visual features and preserving the original source image structure. We also report edit efficiency, defined as ΔCLIP-T per unit LPIPS-S, to summarize this tradeoff.

Table 1 shows that the residualized concatenated model provides the best overall balance. Although Resid+NoConcat obtains the largest CLIP-T and ΔCLIP-T among SAE variants, it also produces substantially higher LPIPS-S, indicating greater source-image distortion. Conversely, NoResid+Concat and Matryoshka slightly reduce LPIPS-S, but achieve smaller target-prompt gains. Resid+Concat achieves near-best target alignment while keeping source distortion low, resulting in the highest edit efficiency among the SAE variants.

5 Related Work

Recent work has established SAEs as a useful tool for interpreting text-to-image diffusion models. SAEs trained on diffusion activations have been shown to learn interpretable features that causally affect generation, evolve over denoising time, and support feature-level interventions (Surkov et al., 2025; Tinaz et al., 2025; Shabalin et al., 2025).

Other work uses SAE features for controllable generation or concept unlearning (Kim and Ghadiyaram, 2025; Cywiński and Deja, 2025), and temporal-aware SAE methods have also been proposed for Diffusion Transformers (Huang et al., 2026). Our work differs by learning sparse dictionaries over residualized activation trajectories rather than isolated timestep activations, timestep-conditioned activations, or non-residualized temporal representations.

Our method also builds on work showing that diffusion activations encode rich spatial and semantic information useful for segmentation, attribution, correspondence, multi-timestep descriptors, and control (Baranchuk et al., 2022; Tang et al., 2023b,a; Zhang et al., 2023; Luo et al., 2023; Hertz et al., 2023; Tumanyan et al., 2023; Epstein et al., 2023). These methods demonstrate the representational and causal value of diffusion activations, while our focus is on learning residualized sparse feature at a trajectory-level.

6 Conclusion

We introduced residualized temporal SAEs for interpreting diffusion activation trajectories. Rather than training directly on raw temporally concatenated activations, we first separate each trajectory into linearly predictable components and residual components using ridge predictors between neighboring timesteps. Training an SAE on this residualized representation encourages sparse latents to capture structure beyond temporally redundant information. The resulting decoder directions can be mapped back into activation space, allowing each latent to be interpreted as a feature trajectory over denoising time. Through reconstruction and ablation studies, spatiotemporal feature analysis, and qualitative steering experiments on Stable Diffusion 1.5, we show that residualized temporal SAEs provide a useful framework for studying temporally structured diffusion activations.

Limitations and Future Work. Our experiments have several limitations. First, we focus on a single Stable Diffusion U-Net activation source. Extending the analysis to other U-Net blocks, model families, and newer architectures would test whether residualized temporal SAEs capture general temporal structure across diffusion models. Second, our steering experiments are meant for qualitative evaluation. Future work can be done to improve steering performance.

Acknowledgments

This work was supported in part by the DARPA Young Faculty Award, the National Science Foundation (NSF) under Grants #2127780, #2319198, #2321840, #2312517, and #2235472, #2431561, the Semiconductor Research Corporation (SRC), the Office of Naval Research through the Young Investigator Program Award, and Grants #N00014-21-1-2225 and #N00014-22-1-2067, Army Research Office Grant #W911NF2410360. Additionally, support was provided by the Air Force Office of Scientific Research under Award #FA9550-22-1-0253, along with generous gifts from Xilinx and Cisco.

References

Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *International Conference on Learning Representations*, 2022.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Chris Olah. Towards monosemanticity: Decomposing language models with dictionary learning. Transformer Circuits Thread, 2023.

Bart Bussmann, Patrick Leask, and Neel Nanda. BatchTopK sparse autoencoders, 2024.

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

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11472–11481, 2022.

Bartosz Cywiński and Kamil Deja. SAeUron: Interpretable concept unlearning in diffusion models with sparse autoencoders. *Proceedings of Machine Learning Research*, 267:11738–11775, 2025.

Dave Epstein, Allan Jabri, Ben Poole, Alexei A. Efros, and Aleksander Holynski. Diffusion self-guidance for controllable image generation. In *Advances in Neural Information Processing Systems*, 2023.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations*, 2023.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851. Curran Associates, Inc., 2020.

Victor Shea-Jay Huang, Le Zhuo, Yi Xin, Zhaokai Wang, Fu-Yun Wang, Yuchi Wang, Renrui Zhang, Peng Gao, and Hongsheng Li. TIDE: Temporal-aware sparse autoencoders for interpretable diffusion transformers in image generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(1):435–443, 2026.

Dahye Kim and Deepti Ghadiyaram. Concept steerers: Leveraging K-sparse autoencoders for test-time controllable generations, 2025.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning. In *Advances in Neural Information Processing Systems*, pages 30233–30249, 2022.

Grace Luo, Lisa Dunlap, Dong Huk Park, Aleksander Holynski, and Trevor Darrell. Diffusion hyperfeatures: Searching through time and space for semantic correspondence. In *Advances in Neural Information Processing Systems*, 2023.

Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241. Springer, 2015.

Christoph Schuhmann, Andreas Köpf, Richard Vencu, Theo Coombes, Benjamin Trom, and Romain Beaumont. LAION-COCO: 600m synthetic captions from LAION2B-EN. LAION Blog, 2022.

Stepan Shabalin, Ayush Panda, Dmitrii Kharlapenko, Abdur Raheem Ali, Yixiong Hao, and Arthur Conmy. Interpreting large text-to-image diffusion models with dictionary learning, 2025.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

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 *Advances in Neural Information Processing Systems*, 2025.

Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. In *Advances in Neural Information Processing Systems*, 2023a.

Raphael Tang, Linqing Liu, Akshat Pandey, Zhiying Jiang, Gefei Yang, Karun Kumar, Pontus Stenetorp, Jimmy Lin, and Ferhan Ture. What the DAAM: Interpreting stable diffusion using cross attention. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5644–5659, Toronto, Canada, 2023b. Association for Computational Linguistics.

Berk Tinaz, Zalan Fabian, and Mahdi Soltanolkotabi. Emergence and evolution of interpretable concepts in diffusion models. In *Advances in Neural Information Processing Systems*, 2025.

Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2023.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Luisa F. Polanía, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. A tale of two features: Stable diffusion complements DINO for zero-shot semantic correspondence. In *Advances in Neural Information Processing Systems*, 2023.

A Implementation Details

A.1 Generation and Dataset

All experiments use Stable Diffusion 1.5 with the Hugging Face model identifier `runwayml/stable-diffusion-v1-5`. Prompts are sampled from `guangyil/laion-coco-aesthetic`, using the `train` split and the `caption` field. We shuffle prompts with seed 42 and shuffle buffer size 10,000, using the

template `{caption}`. We use 50,000 generated images for the training dataset and 2,500 generated images for the validation dataset. Images are generated at 512×512 resolution with 50 sampling steps, classifier-free guidance scale 7.5, and an empty negative prompt.

Existing assets and licenses. We use Stable Diffusion 1.5 through the Hugging Face model identifier `runwayml/stable-diffusion-v1-5`, which is distributed under the CreativeML OpenRAIL-M license and associated use restrictions. We use only the `caption` field of `guangyil/laion-coco-aesthetic` for prompts; this dataset is distributed on Hugging Face under the Apache-2.0 license and is derived from LAION-COCO synthetic captions for publicly available web images. All uses of these assets should comply with their upstream licenses and terms.

A.2 Activation Collection

We collect activations from the conditional branch of classifier-free guidance. The unconditional branch is computed during generation but is not saved. The main activation source is `mid.0.1`. We save the residual contribution of this module, defined as the module output minus its input.

For stride $\ell = 10$, each subsampled activation has shape $8 \times 8 \times 1280$, corresponding to $S = 64$ spatial tokens and channel dimension $d = 1280$. We use $i = 0, \ldots, T-1$ to index the subsampled timestep blocks and τ_i to denote the corresponding full sampling-step index. In the main experiments, $(\tau_0, \tau_1, \tau_2, \tau_3, \tau_4) = (0, 10, 20, 30, 40)$, giving $T = 5$ subsampled timestep blocks; the terminal sampling-step index 50 is dropped. Each generated image therefore contributes 64 spatial-token trajectories, so the training dataset contains $50{,}000 \times 64 = 3{,}200{,}000$ token trajectories, and the validation dataset contains $2{,}500 \times 64 = 160{,}000$ token trajectories.

For residualized models, we fit ridge predictors between neighboring subsampled timestep indices. The implementation fits one ridge predictor per adjacent subsampled transition $i - 1 \to i$, shared across all training samples and spatial positions. For each transition, the ridge weight has shape 1280×1280 and the bias has shape 1280. The ridge penalty is 0.1. The bias is handled by centering and is not regularized.

A.3 SAE Inputs and Training

For residualized concatenated models, the SAE input is

$$\mathbf{z}_q = [\mathbf{a}_{q,0}, \mathbf{r}_{q,1}, \ldots, \mathbf{r}_{q,T-1}]. \tag{11}$$

For non-residualized concatenated models, residual blocks are replaced by raw activation blocks:

$$\mathbf{z}_q = [\mathbf{a}_{q,0}, \mathbf{a}_{q,1}, \ldots, \mathbf{a}_{q,T-1}]. \tag{12}$$

With stride 10 and five subsampled timestep blocks, the concatenated input dimension is $5 \times 1280 = 6400$. For non-concatenated models, we train one SAE per timestep block, so each timestep-wise SAE has input dimension 1280.

All main experiments use BatchTopK sparse autoencoders. The SAE uses untied encoder and decoder weights, encoder and decoder biases, a ReLU before BatchTopK selection, Kaiming initialization, and decoder row normalization after each optimizer step. The auxiliary dead-latent loss weight is $1/32$, the dead-latent threshold is 2000, and the auxiliary top-k value is 256. We train with Adam using learning rate 10^{-4}, batch size 256, and 30 epochs. The SAE training seed is 43, and checkpoints are selected by best validation explained variance.

Latent dictionary sizes are allocated relative to the full trajectory dimension. For example, at expansion factor 0.5 with stride 10, the full trajectory dimension is $5 \times 1280 = 6400$, giving 3200 total latents. For timestep-wise models, this corresponds to 640 latents for each of the five timestep-specific SAEs.

Matryoshka SAE baseline. We additionally evaluate a Matryoshka SAE baseline in which latent groups are aligned with temporal chunks of a concatenated activation trajectory. This baseline is trained on non-residualized concatenated activation trajectories. Each timestep component is normalized independently using statistics computed on the training split, and the normalized components are then concatenated:

$$\mathbf{x}_q = [\tilde{\mathbf{a}}_{q,0}, \tilde{\mathbf{a}}_{q,1}, \ldots, \tilde{\mathbf{a}}_{q,T-1}] \in \mathbb{R}^{Td}, \tag{13}$$

where $\tilde{\mathbf{a}}_{q,i}$ denotes the normalized activation chunk for token trajectory q at subsampled timestep index i. Thus, this baseline uses the same trajectory-level concatenation structure as the main concatenated variants, but does not replace later timestep chunks with ridge residuals.

The Matryoshka SAE uses a single encoder and decoder, but partitions the latent dimension into contiguous groups

$$\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_{T-1}, \tag{14}$$

with one latent group per subsampled timestep chunk. Given sparse latent activations $\mathbf{h}_q$, the model forms a sequence of nested prefix reconstructions

$$\mathbf{x}^{\mathrm{rec}}_{q,-1} = \mathbf{b}_{\mathrm{dec}}, \tag{15}$$

$$\mathbf{x}^{\mathrm{rec}}_{q,j} = \mathbf{b}_{\mathrm{dec}} + \sum_{i=0}^{j} W_{\mathrm{dec},\mathcal{G}_i} \mathbf{h}_{q,\mathcal{G}_i}, \qquad j = 0, \ldots, T-1. \tag{16}$$

Each prefix reconstruction is trained to reconstruct the full concatenated trajectory rather than only the corresponding timestep block. The reconstruction objective averages the full-trajectory MSE over all prefixes:

$$\mathcal{L}_{\mathrm{mat}} = \frac{1}{T+1} \left(\left\| \mathbf{x}_q - \mathbf{x}^{\mathrm{rec}}_{q,-1} \right\|_2^2 + \sum_{j=0}^{T-1} \left\| \mathbf{x}_q - \mathbf{x}^{\mathrm{rec}}_{q,j} \right\|_2^2 \right) + \beta_{\mathrm{aux}} \mathcal{L}_{\mathrm{aux}}. \tag{17}$$

The auxiliary loss $\mathcal{L}_{\mathrm{aux}}$ is the same dead-latent recovery loss used for the other BatchTopK SAE variants.

Sparsity is enforced globally across all latent groups, not separately within each group. For a minibatch of size B, BatchTopK retains approximately BK_{avg} active latents across the full grouped latent vector. Thus, the Matryoshka baseline has temporally grouped latents, but its sparsity budget is shared across all groups and its input is a non-residualized, component-normalized activation trajectory rather than a residualized trajectory.

In our experiments, trajectories are subsampled every 10 DDIM steps. For evaluation, we follow the same chunk convention as the other variants and drop the final terminal chunk, so reported reconstruction metrics are computed over the subsampled timestep chunks with full sampling-step indices $\{0, 10, 20, 30, 40\}$. Each activation chunk has dimension $d = 1280$. We use an expansion factor of 0.5 per group, giving 640 latents per timestep group. Unless otherwise specified, we use $K_{\mathrm{avg}} = 64$, auxiliary-loss weight $\beta_{\mathrm{aux}} = 1/32$, a dead-latent threshold of 2000 steps, at most 256 auxiliary latents, AdamW with learning rate 10^{-4}, batch size 256, and 30 training epochs.

A.4 Evaluation

All reconstruction metrics are computed on the validation dataset and evaluated in the original activation space, not in normalized SAE input space. For residualized models, the first activation block is reconstructed directly from the SAE output, and later residual blocks are mapped back to activation space using teacher forcing:

$$\mathbf{a}^{\mathrm{rec}}_{q,i} = W_i \mathbf{a}_{q,i-1} + \mathbf{b}_i + \mathbf{r}^{\mathrm{rec}}_{q,i}, \qquad i = 1, \ldots, T-1. \tag{18}$$

The ground-truth activation $\mathbf{a}_{q,i-1}$ is used as input to the ridge predictor. Thus, reconstruction evaluates how well the SAE reconstructs the residual components at each subsampled timestep index, rather than autonomous rollout dynamics.

We report mean squared error, explained variance, and per-timestep explained variance:

$$\mathrm{MSE} = \frac{1}{MTd} \sum_{q=1}^{M} \sum_{i=0}^{T-1} \left\| \mathbf{a}_{q,i} - \mathbf{a}^{\mathrm{rec}}_{q,i} \right\|_2^2, \tag{19}$$

$$\mathrm{EV} = 1 - \frac{\sum_{q,i} \left\| \mathbf{a}_{q,i} - \mathbf{a}^{\mathrm{rec}}_{q,i} \right\|_2^2}{\sum_{q,i} \left\| \mathbf{a}_{q,i} - \boldsymbol{\mu}_a \right\|_2^2}, \qquad \mathrm{EV}_i = 1 - \frac{\sum_{q} \left\| \mathbf{a}_{q,i} - \mathbf{a}^{\mathrm{rec}}_{q,i} \right\|_2^2}{\sum_{q} \left\| \mathbf{a}_{q,i} - \boldsymbol{\mu}_{a,i} \right\|_2^2}. \tag{20}$$

Here M is the number of token trajectories, d is the activation dimension, $\boldsymbol{\mu}_a$ is the mean activation over the evaluation set, and $\boldsymbol{\mu}_{a,i}$ is the mean activation at subsampled timestep index i.

For timestep-wise models, we report two sparsity-matching protocols. In the trajectory-level matched setting used in the main text, the total average number of active latents across the T timestep-wise SAEs is matched to the average number of active latents used by a concatenated trajectory SAE. In the per-timestep-matched setting reported in Appendix B, the same average sparsity is applied independently to each timestep-wise SAE, so these models have a larger total trajectory-level sparse-code budget than concatenated models with the same per-block sparsity.

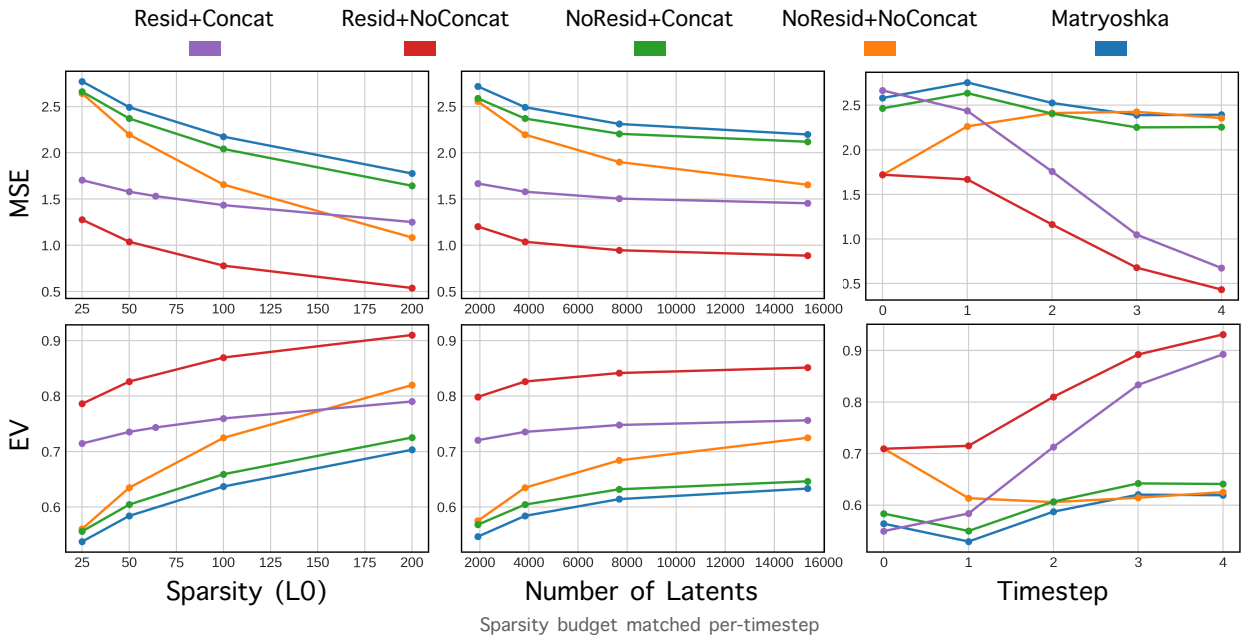

Figure 5: Activation-space reconstruction for BatchTopK SAE variants under per-timestep sparsity-budget matching. Overall MSE/EV are shown versus sparsity and dictionary size; per-timestep curves use the representative configuration $K_{\text{avg}} = 50$ and expansion factor 0.5. In this setting, each timestep-wise SAE uses the same average sparsity as the concatenated trajectory SAE, giving timestep-wise baselines an effective trajectory-level sparse-code budget of approximately TK_{avg}. The residualized timestep-wise model therefore serves as a reconstruction-favorable baseline. The Matryoshka curve is a non-residualized, component-normalized Matryoshka baseline.

A.5 Compute Resources

Experiments were run on an internal GPU server with 8 NVIDIA A100-SXM4-80GB GPUs, AMD EPYC 7713 CPUs, and approximately 2.1 TB RAM. Each training run used a single GPU, with multiple runs scheduled across available GPUs.

The main SD1.5/COCO activation dataset uses 50,000 training prompts and 2,500 validation prompts. The full activation cache occupies approximately 409 GiB, while the stride-10 cache used for the main experiments occupies approximately 48 GiB. The derived SAE input data for each model family occupies approximately 92 GiB.

Activation generation and extraction required about 11 A100 GPU-hours. Preprocessing required under 1 GPU-hour. The completed hyperparameter runs used for the reported BatchTopK results required approximately 672 A100 GPU-hours, for an estimated total of about 700 A100 GPU-hours.

B Additional Reconstruction Results

Per-timestep sparsity-budget matching. Figure 5 reports the reconstruction comparison under per-timestep sparsity-budget matching. In this setting, the same BatchTopK average sparsity K_{avg} is used for the concatenated SAE and for each timestep-wise SAE. As a result, timestep-wise models use approximately TK_{avg} active latents over a full token trajectory, while concatenated models use K_{avg} active latents for the entire trajectory. This gives timestep-wise baselines a larger effective trajectory-level sparse-code budget, so this comparison should be interpreted as a reconstruction-favorable setting for timestep-wise models rather than as a matched-budget comparison.

Even under this favorable setting for timestep-wise reconstruction, residualization improves reconstruction relative to the corresponding non-residualized variants. The strongest reconstruction performance is achieved by the residualized timestep-wise baseline, as expected from its larger total sparse-code budget. The more direct trajectory-level comparison remains Resid+Concat versus NoResid+Concat, where residualization consistently improves activation-space reconstruction while preserving a single sparse code for the full denoising trajectory.

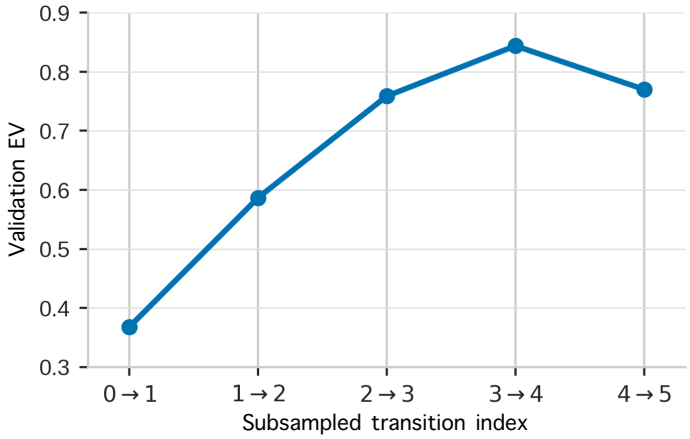

Figure 6: Validation explained variance of the ridge predictors used for linear residualization. Each point corresponds to one adjacent transition between subsampled timestep indices for the main activation source. A simple linear predictor explains a substantial fraction of activation variance at each transition, indicating strong temporal redundancy and motivating the residualized trajectory representation.

C Additional Residualization Details

To verify that residualization removes a meaningful source of temporal redundancy, we measured the validation explained variance of the ridge predictors used in Section 3.2. For each adjacent subsampled transition, we fit the ridge predictor on the training split and evaluated prediction quality on validation activation trajectories. Figure 6 shows that a simple linear predictor explains a substantial fraction of the variance across subsampled timestep indices, with explained variance ranging from approximately 0.37 to above 0.8. This indicates that neighboring diffusion activations contain strong linearly predictable structure. Residualization therefore removes predictable timestep-to-timestep drift before SAE training, encouraging the sparse dictionary to model residual components rather than reconstructing redundant temporal structure.

D Additional Feature Analysis

D.1 Highest activating samples per feature

Figure 7 shows the highest-activating generated samples for selected SAE latents. Each row corresponds to one latent, and columns show the samples with the largest activation scores for that latent. The examples show that many temporal SAE latents are associated with coherent visual themes, including food and bowls, people and clothing, paired human subjects, object categories, sports scenes, furniture, vehicles, houses, pants, clocks, and children.

D.2 Temporal self-similarity

For each latent k, we summarize the temporal stability of its activation-space decoder trajectory by the mean off-diagonal cosine similarity

$$s_k = \frac{1}{T(T-1)} \sum_{i \neq j} \cos\left(\phi_{k,i}, \phi_{k,j}\right). \tag{21}$$

Figure 8 shows the distribution of s_k for early, middle, and late temporal groups. Early features have higher cross-timestep self-similarity than middle and late features, suggesting that early features correspond to more temporally persistent activation-space directions, while later features are more timestep-specific.

D.3 Spatial localization statistics

For a cosine-similarity spatial mask $c_{k,i}^{(u,v)}$, we compute positive spatial entropy by first normalizing the positive part of the mask,

$$p_{k,i}^{+}(u,v) = \frac{\max(c_{k,i}^{(u,v)}, 0)}{\sum_{u',v'} \max(c_{k,i}^{(u',v')}, 0) + \epsilon}, \tag{22}$$

and then computing entropy normalized by the maximum possible entropy on the $S = HW$ spatial grid,

$$H_{k,i}^{+} = -\frac{1}{\log S} \sum_{u,v} p_{k,i}^{+}(u,v) \log \left(p_{k,i}^{+}(u,v) + \epsilon \right). \tag{23}$$

Figure 9 shows that positive spatial entropy decreases from early to middle and late temporal groups, indicating that early features tend to have broader positive spatial support while later features are more spatially concentrated. Averaging the mean entropy values over the five subsampled timestep indices gives 0.876 for early features, 0.835 for middle features, and 0.772 for late features. The corresponding median ranges across subsampled timestep indices are 0.878–0.971 for early features, 0.862–0.927 for middle features, and 0.812–0.847 for late features.

D.4 Additional spatial localization maps

Figure 10 shows additional spatial localization maps for representative early, middle, and late temporal features. For each feature, we show the highest-activating generated sample together with cosine-similarity maps between the feature's activation-space decoder direction and the spatial U-Net tokens across subsampled timestep indices. These examples complement the aggregate spatial-entropy results in Figure 9. Early features often exhibit broad spatial support, including diffuse or border-like masks. Middle features more often produce structured masks concentrated on salient regions or object-like areas. Late features tend to be more spatially variable and localized, consistent with the lower positive spatial entropy observed for the late temporal group.

E Additional Steering Results

E.1 Additional single-feature steering examples

Figure 11 provides additional qualitative steering examples for the representative early, middle, and late features used in the main text. For each feature, we compare local steering, where the intervention is applied only to a selected spatial region, with global steering, where the same activation-space direction is applied to all spatial tokens. The examples show that temporal SAE directions can induce visible changes in generated images, with stronger interventions producing larger semantic and textural shifts. Local steering can concentrate the effect in a restricted image region at lower strengths, while global steering more readily changes the overall image appearance. These examples are intended as qualitative evidence that the learned activation-space decoder trajectories have visible generative effects under intervention, not as optimized image-editing results.

E.2 Implementation details for feature transfer steering

Masking and feature selection. For the RIEBench evaluation, we use the grounding phrases provided with each example to obtain source and target object masks. Specifically, we apply Grounded-SAM2 separately to the unsteered source and target generations, using the source grounding phrase for the source image and the target grounding phrase for the target image. These masks define the spatial regions used for both feature selection and feature transfer.

Let $\mathbf{h}^{\mathrm{src},(u,v)}, \mathbf{h}^{\mathrm{tgt},(u,v)} \in \mathbb{R}^m$ denote the source and target encoded tensor at spatial location (u,v), defined for $H \times W$ spatial token trajectories. Also let $\mathcal{M}^{\mathrm{src}}$ and $\mathcal{M}^{\mathrm{tgt}}$ denote the spatial masks for the source and target generations, respectively. These masks are obtained by applying Grounded-SAM2 to the unsteered source and target images using the corresponding RIEBench grounding phrases.

For each candidate SAE feature k, we first compute its masked average activation over the source and target trajectories:

$$\bar{c}_{k,i}^{\mathrm{src}} = \frac{1}{|\mathcal{M}^{\mathrm{src}}|} \sum_{(u,v) \in \mathcal{M}^{\mathrm{src}}} h_k^{\mathrm{src},(u,v)}, \tag{24}$$

$$\bar{c}_{k}^{\mathrm{tgt}} = \frac{1}{|\mathcal{M}^{\mathrm{tgt}}|} \sum_{(u,v) \in \mathcal{M}^{\mathrm{tgt}}} h_k^{\mathrm{tgt},(u,v)}. \tag{25}$$

We then rank features by the masked contrast score

$$\gamma_k = \frac{\bar{c}_k^{\mathrm{src}}}{\sum_j \bar{c}_j^{\mathrm{src}}} - \frac{\bar{c}_k^{\mathrm{tgt}}}{\sum_j \bar{c}_j^{\mathrm{tgt}}}. \tag{26}$$

The transferred feature set is the top-p features under this score:

$$\mathcal{K} = \mathrm{Top}_p \left(\{\gamma_k\}_{k=1}^{d_{\mathrm{SAE}}} \right). \tag{27}$$

For non-concatenated temporal SAE models, which use separate SAEs across temporal chunks, we select the top features using the first temporal chunk and reuse the same local feature indices for all later chunks.

Steering hyperparameters. All source-to-target steering interventions are applied only to the conditional branch of classifier-free guidance; the unconditional branch is left unchanged. We use the same classifier-free guidance scale as the base Stable Diffusion generation, 7.5. We use a steering strength of $\alpha = 10$, and set $p = 50$ for the top-p features.

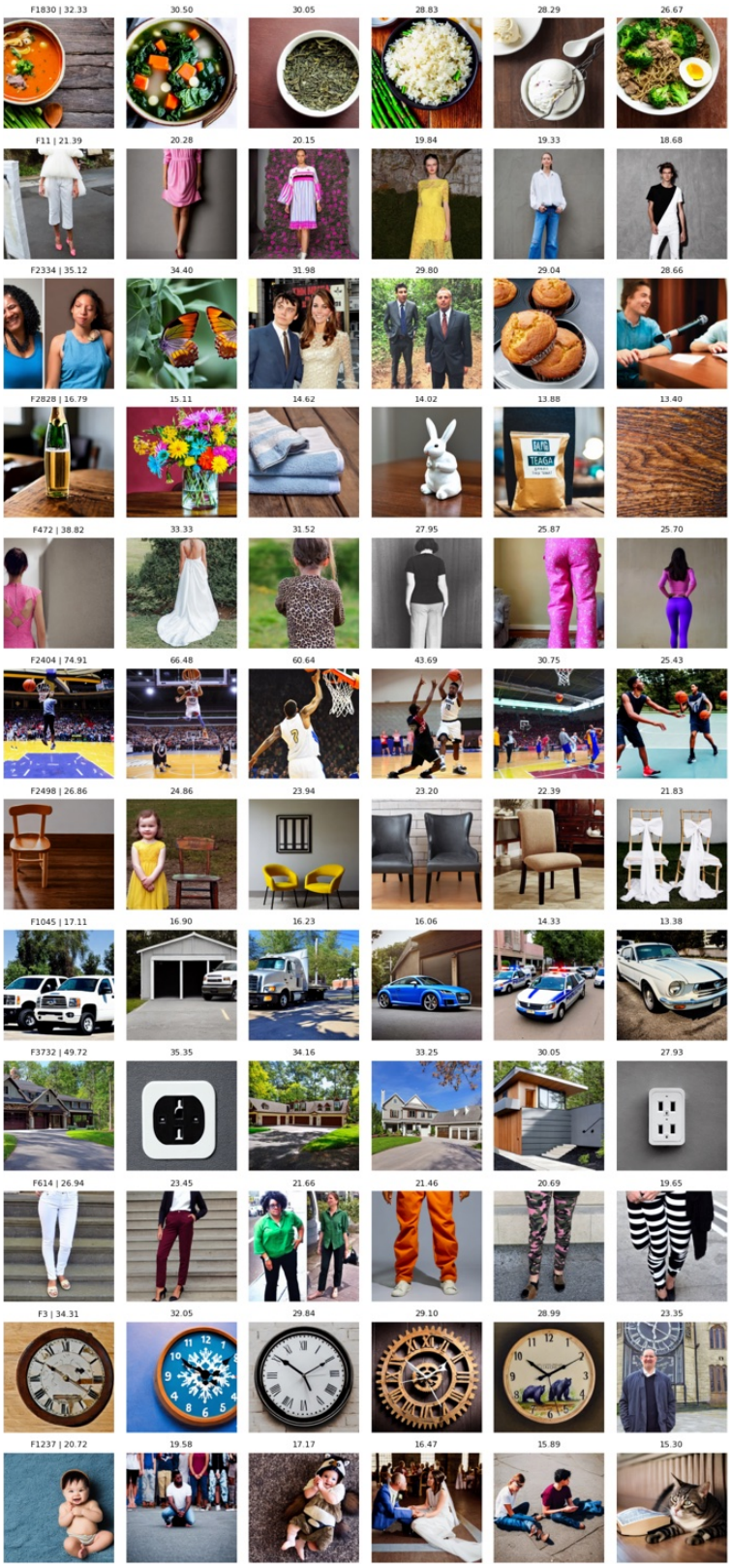

Figure 7: Highest-activating generated samples for selected temporal SAE latents. Each row corresponds to one latent, labeled by its feature index, and each column shows one of the samples with the largest activation score for that latent. The number above each image denotes the corresponding activation score. The rows illustrate that individual latents often retrieve semantically coherent sets of images. Samples are selected by maximum latent activation over spatial tokens and subsampled timestep indices on the validation set.

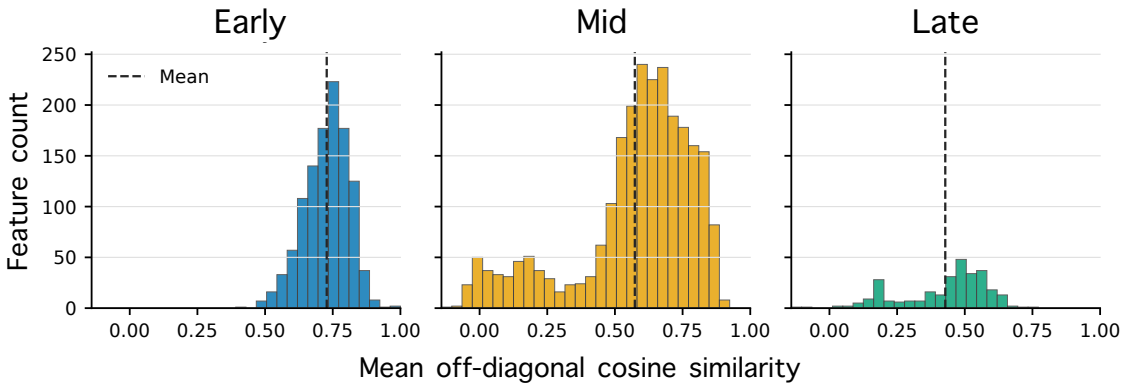

Figure 8: Temporal self-similarity by feature class. For each latent, we compute the mean off-diagonal cosine similarity between activation-space decoder directions across subsampled timestep indices. Higher values indicate that a feature direction is more stable across denoising time.

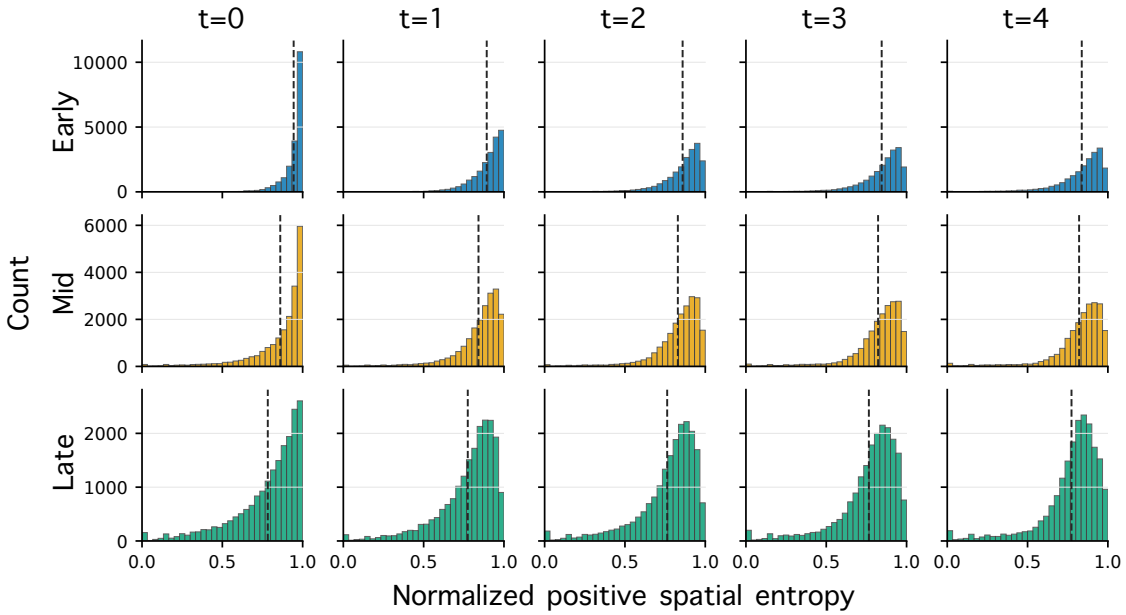

Figure 9: Positive spatial-mask entropy by temporal profile class and subsampled timestep index. Entropy is computed from the ReLU-normalized cosine-similarity mask and divided by $\log S$, where S is the number of spatial tokens. Higher entropy indicates broader positive spatial support.

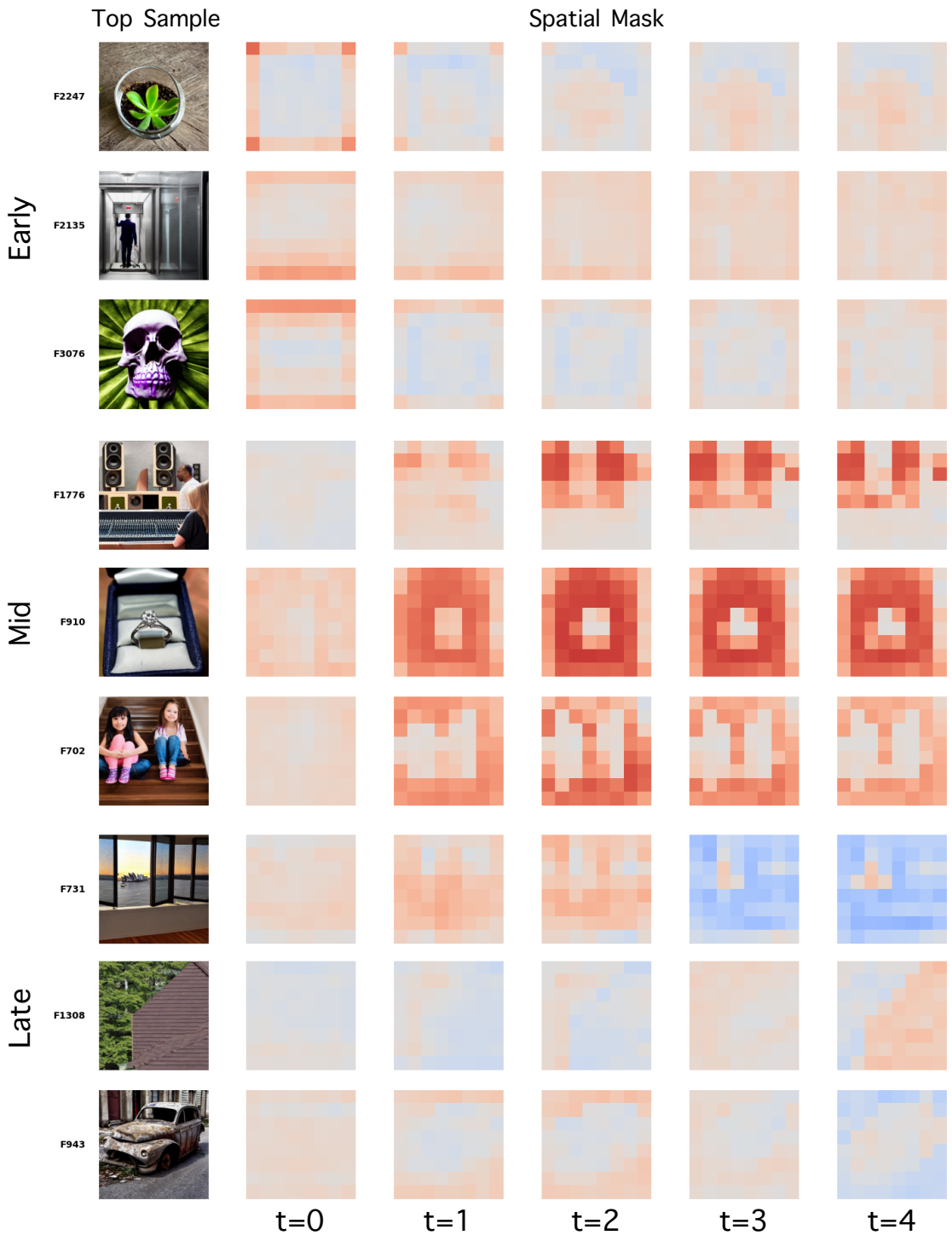

Figure 10: Additional spatial localization maps for representative early, middle, and late temporal features. For each feature, we show the highest-activating sample and the corresponding spatial cosine-similarity masks across subsampled timestep indices. Early features often have broad or border-like support, whereas middle and late features tend to produce more localized or structured spatial masks.

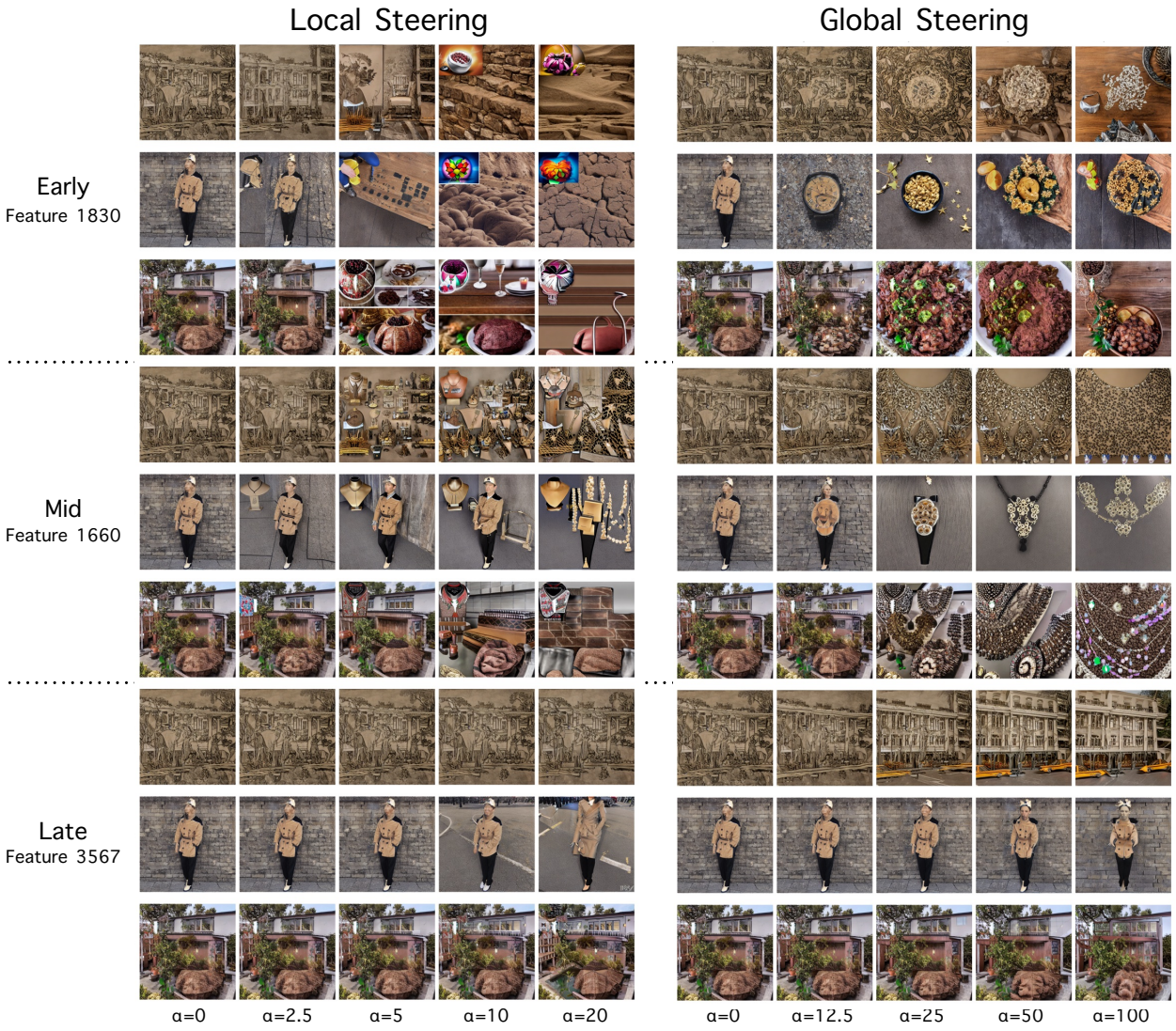

Figure 11: Additional qualitative steering examples for representative early, middle, and late temporal SAE features. For each feature, we compare local steering at strengths $\alpha \in \{0, 2.5, 5, 10, 20\}$ with global steering at strengths $\alpha \in \{0, 12.5, 25, 50, 100\}$. Local steering applies the activation-space feature direction only to a selected spatial region, while global steering applies it to all spatial tokens. Increasing α generally strengthens the visual effect, with global interventions producing broader image-level changes.