Revelio: Interpreting and leveraging semantic information in diffusion models

Dahye Kim[1*] Xavier Thomas[1*] Deepti Ghadiyaram[12†]
[1]Boston University [2]Runway
{dahye, xthomas, dghadiya}@bu.edu

Abstract

We study how rich visual semantic information is represented within various layers and denoising timesteps of different diffusion architectures. We uncover monosemantic interpretable features by leveraging k-sparse autoencoders (k-SAE). We substantiate our mechanistic interpretations via transfer learning using light-weight classifiers on off-the-shelf diffusion models' features. On 4 datasets, we demonstrate the effectiveness of diffusion features for representation learning. We provide an in-depth analysis of how different diffusion architectures, pre-training datasets, and language model conditioning impacts visual representation granularity, inductive biases, and transfer learning capabilities. Our work is a critical step towards deepening interpretability of black-box diffusion models. Code and visualizations available at: https://github.com/revelio-diffusion/revelio

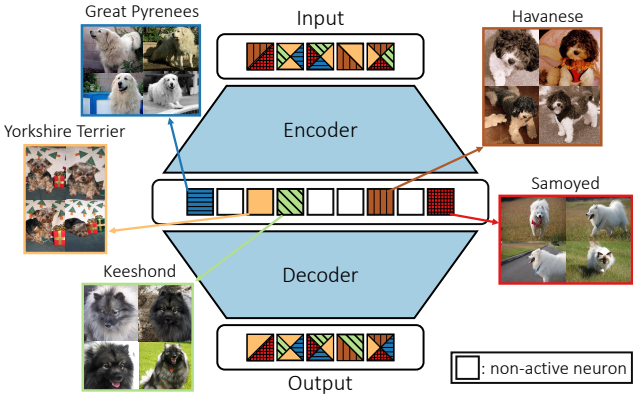

Figure 1. **k-sparse autoencoders (k-SAE)** trained on complex visual features help identify monosemantic visual properties represented within black-box diffusion models. We show sample k-SAE neurons and top-4 images that yield highest activations when the k-SAE is trained on intermediate diffusion layer's features on Oxford-IIIT Pet [41] dataset. Note how these features encapsulate distinct fine-grained information about different breeds like *Keeshond* and *Samoyed*. Best viewed in color.

1. Introduction

Generating high-quality photo-realistic and creative visual content using diffusion models is a thriving area of research. For a generative model to accurately simulate the visual world around us, its latent space should in principle capture rich visual semantics and the physical dynamics of the real world. A direct empirical evidence is in recent efforts that leverage diffusion features for discriminative tasks such as detection [11], segmentation [9, 64], classification [30], semantic correspondence [33], depth estimation [63, 68], or visual reasoning [62] tasks. Yet, they do not offer clear insights on *how* this rich semantic information is represented within the model. Some prior attempts that visualize attention maps [7] or use PCA [60] on the intermediate features, though valuable, operate on per-image basis and thus, do not offer a more holistic in-depth interpretation of diffusion models' internal representations.

In this work, we go beyond harnessing the rich representations of diffusion models and aim to fundamentally understand and interpret diffusion models' internal states. Concretely, we address the following questions: what flavors of visual information is captured in different layers and time-steps of a diffusion model? How do they interact with and complement each other and the overall learnt visual information? Do different layers benefit differently from external conditioning and why? What inductive biases are uniquely captured in convolution-based diffusion models compared to transformer-based ones?

Understanding how a model learns visual information offers several key benefits. First, current visual generative models are black box in nature: it is not clear why a benign prompt sometimes produces an unsafe output or why a very slight tweak to the same prompt generates a very different output [5]. Answering the above fundamental questions will be a crucial step towards interpreting black box generative models. Second, distilling the granularity of semantic information represented across different layers, timesteps, and model architectures can aid in designing more efficient algorithms that offer semantic and style control.

To *reveal* the visual knowledge learnt by diffusion models, we adopt "mechanistic interpretation" techniques and

*Equal contribution.
†Corresponding author.

learn a sparse dictionary of monosemantic visual concepts. It is physiologically proven that human visual system *sparsely* encodes the most recurring visual patterns using a small set of basis functions [39]. Motivated by this, we aim to uncover interpretable features by leveraging k-sparse auto-encoders (k-SAE) [36], which have been shown to help interpret language models [14, 56]. We illustrate *how* the semantic visual information is packed differently depending on the representation granularity of the test dataset, across different diffusion layers, denoising timesteps, model architectures, and pre-training data.

Going beyond this, we corroborate our mechanistic interpretation by learning very light-weight classifiers on top of off-the-shelf diffusion models' features. Through rigorous analysis against multiple baselines and benchmarks, we show the surprising effectiveness of diffusion features across a variety of tasks: coarse and fine-grained classification and complex visual reasoning. Unlike all prior works, our classifier, dubbed, Diff-C, bypasses the need to employ additional losses [30], training a student model [65], or training a feature map fusing method [33], thereby offering significant computational benefits (**4** orders of magnitude inference speedup compared to [30][1]). We summarize our empirical and interpretable analysis below, which align perfectly across datasets, tasks, and model architectures.

- **Representation granularity varies non-linearly with model depth**, with different diffusion layers capturing varying levels of visual semantic information, from coarse-grained shape, texture, or local color patterns to fine-grained animal breed details, to more global visual concepts like camera angles and object poses.
- **Representation granularity and generalizability varies with diffusion architectures, pre-training data, latent or pixel space, cross and self-attention mechanisms** – design choices made to improve the overall pixel generation quality and training efficiency.
- **k-sparse autoencoders help isolate monosemantic visual properties** systematically across model states and help interpret black box diffusion models.

2. Related Work

Diffusion features for discriminative tasks: Diffusion models have achieved remarkable results in generating semantically rich high-resolution images [8, 17, 25, 43, 48, 50]. Several recent works leverage diffusion features beyond image and video synthesis: for zero-shot classification [13, 30, 65], detection [11], segmentation [9, 64], semantic correspondence [24, 33], rendering novel views [66], image editing and semantic image manipulation tasks [29, 60], and so on. Our work is different from prior works in two important ways. First, we propose a simple method to *adapt* diffusion features for discriminative tasks without the need to distill [65], train an expensive hypernetwork [33], or generate synthetic data [26]. Second, we go beyond leveraging diffusion features and interpret *how* visual information is packed with the model's architecture.

Interpreting diffusion features: Some recent studies aim at understanding and interpreting diffusion models. Plug-and-Play [60] performs PCA analysis on intermediate features of Stable Diffusion [48] and finds that intermediate features reveal localized semantic information shared across objects, while early layers capture high-frequency details. However, their analysis is based only on 20 real and generated humanoid images, limiting the generalizability of their findings to different domains and model architectures. Authors of [22] explore how diffusion features vary with the underlying architecture. Similarly, the effect of cross-attention layers to image attributes [61] and semantic information at different timesteps [34, 42] have been studied by progressively conditioning text prompts. While valuable, these analyses are also done on a per-image basis and do not offer a holistic and in-depth interpretation of the models' internal states. Diffusion Lens [58] analyzes the text encoder of diffusion models by generating images from its intermediate representations. By contrast, our work mechanistically interprets the opaque *visual* diffusion features when conditioned on blank prompts using sparse autoencoders.

Recent works have demonstrated that sparse autoencoders (SAE) could recover monosemantic features in large language models (LLMs) [10, 14, 20] and CLIP vision features [15, 19]. Concurrent work [54] investigates the possibility of using SAEs to learn interpretable features from residual updates within the U-Net to investigate how the cross-attention layer integrates the input text prompt. By contrast, our focus is to understand how visual information is packed within the diffusion models' internal states and the interplay between representation abstraction and model design choices. We propose a method to reveal valuable human interpretable visual patterns baked within black box diffusion models.

3. Approach

Our goal is to interpret and expand our understanding of black box diffusion models. We address this from two different perspectives: first, we train k-sparse autoencoders [36] to recover interpretable monosemantic visual semantic features across different layers, timesteps, and diffusion architectures. Second, we substantiate each interpretability finding by training light-weight classifiers on the exact same diffusion features.

We begin by providing an overview of diffusion models (Sec. 3.1), followed by motivation and architecture used for training k-SAE (Sec. 3.2), followed by the light-weight classifier, **Diff-C** (Sec. 3.3).

[1] Our method requires training a classifier to achieve this.

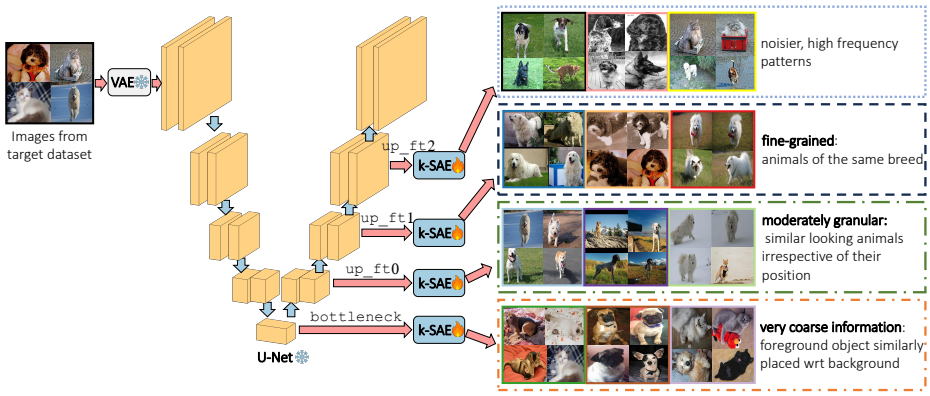

Figure 2. **k-SAE visualizations across layers of the U-Net** in SD 1.5 and sample images from different neurons yielding highest activations when k-SAEs are trained on different layers for $t = 25$ on Oxford-IIIT Pet. We note that across 3 random neurons of k-SAEs, the `bottleneck` layer captures very coarse-grained information, where foreground objects positioned similarly are activated by the same neuron. `up_ft1` captures valuable breed specific domain information while `up_ft2` seems to capture high-frequency visual patterns.

3.1. Preliminaries on diffusion models

Several powerful open and close-sourced diffusion models have emerged just in the last two years [1, 2, 21, 43, 45, 47, 48, 50]. Broadly, diffusion models are probabilistic generative models that aim to learn a data distribution $p(x)$ through an iterative denoising process. During the forward diffusion process, the input image x is gradually perturbed with noise over T timesteps. The reverse process consists of iterative denoising steps, where each step estimates the added noise $\epsilon_\theta(x_t, t)$, parameterized by θ, with $t = 1, \ldots, T$. Each iteration takes a noisy image x_t as input and predicts the added noise ϵ. The objective of the diffusion model is given by:

$$L_{DM} = \mathbb{E}_{x,t,\varepsilon\sim\mathcal{N}(0,1)} \left[\|\varepsilon - \varepsilon_\theta(x_t, t)\|_2^2 \right] \quad (1)$$

Instead of operating on images x, latent diffusion models (LDM) [48] operate on a latent representation z, obtained by mapping the image into a lower-dimensional space using a variational autoencoder [27] which consists of an encoder $\mathcal{E}$ and decoder $\mathcal{D}$. The diffusion process models the distribution of these latent embeddings, allowing for more efficient computation. The revised objective is:

$$L_{LDM} = \mathbb{E}_{\mathcal{E}(x),t,\varepsilon\sim\mathcal{N}(0,1)} \left[\|\varepsilon - \varepsilon_\theta(z_t, t)\|_2^2 \right] \quad (2)$$

3.2. Preliminaries on k-sparse autoencoders

Our aim is to gain insight into how visual information is encapsulated in a diffusion model. Given the very high non-linearity and complex architectures of generative models, identifying interpretable components directly from layer activations is not viable. In this work, we isolate monosemantic features by training k-sparse autoencoders (k-SAEs) on the activations from different diffusion layers, timesteps, and architectures, which we describe next.

Sparse autoencoders [37] are neural networks to learn compact feature representations in an unsupervised manner. They contain an encoder and a decoder, and are trained with a sparsity penalty and a sample reconstruction loss to encourage only a few neurons to be maximally activated for a given input. However, the sparsity penalty term in SAEs presents significant training challenges [36, 57]. A k-sparse autoencoder is an extension of sparse autoencoder [37] designed to improve the training challenges by explicitly regulating the number of active neurons during training to k. Specifically, in each training step, a top-k activation function is used to retain only the k largest neuron activations, while zeroing out the rest.

Let x denote the d-dimensional spatially-pooled diffusion activations[2]. Let $W_{enc} \in \mathbb{R}^{n\times d}$ and $W_{dec} \in \mathbb{R}^{d\times n}$ denote the weight matrices of the k-SAE's encoder and decoder respectively (Fig. 1), where n denotes the dimension of the autoencoder's hidden layer. n is equal to d multiplied by a positive integer, called the expansion factor. Following [10], $b_{pre} \in \mathbb{R}^d$ denotes the bias term added to input x before feeding to the encoder (aka pre-encoder bias), while $b_{enc} \in \mathbb{R}^n$ denotes the bias term for encoder. Upon passing x through the encoder, we obtain z defined as:

$$z = \text{TopK}(W_{enc}(x - b_{pre}) + b_{enc}), \quad (3)$$

where the TopK activation function retains only the top k neuron activations and sets the rest to zero [36]. The decoder then reconstructs z, given by:

$$\hat{x} = W_{dec} z + b_{pre} \quad (4)$$

The training loss is the normalized reconstruction mean squared error (MSE) between the reconstructed feature ($\hat{x}$) and the original feature (x), given by:

$$L_{mse} = \|x - \hat{x}\|_2^2 \quad (5)$$

[2]For notational simplicity, we describe our setup for an arbitrary layer and denoising timestep, but the same method applies for activations from any model state.

As we show in results (Sec. 4), k-SAE plays a key role in qualitatively interpreting visual semantic information.

3.3. Diffusion Classifier (Diff-C)

Next, to quantitatively study the visual semantic information packed within pre-trained diffusion models, we design a lightweight classifier called **Diff-C** to adapt diffusion features to downstream tasks. **Diff-C** (shown in Table 10 in suppl. material) comprises a series of convolutional layers to progressively reduce the spatial dimensions of the diffusion features, followed by a pooling and a downstream task-dependent fully-connected layer. Despite the inherently unique architectures of convolutions-based U-Net [49] and diffusion-based DiT [43], we adapt the outputs of different U-Net layers and DiT blocks into 2D feature maps and process them using Diff-C.

4. Experiments

In this section, we first share the implementation details and training setup, followed by detailed analyses while dissecting diffusion models.

4.1. Implementation details

Unless otherwise stated, we use Stable Diffusion (SD) 1.5 model [48], DDIM scheduler [53], and an empty prompt used as the text conditioning.

k-SAE: For training a k-SAE, we empirically found that $k = 32$ yields the best results for different datasets, based on training stability and overall sparsity. Diffusion activations are extracted by passing images from a target dataset into the VAE of a pretrained diffusion model. For SD, we spatially pool diffusion activations resulting in $d = 1280$ for `bottleneck`, `up_ft0`, and `up_ft1` layers. We set the expansion factor for the k-SAE to 64, following prior work [19], resulting in $n = 1280 \times 64 = 81,920$ latents for SD and $n = 1152 \times 64 = 73,728$ latents for DiT. We apply a unit normalization constraint [51] on the decoder weights W_{dec} of the k-SAE after each update. We use the Adam [28] optimizer with a learning rate of 0.0004 and apply a constant warm up for 500 steps. The total training time approximately 1 hour on 1 NVIDIA RTX A6000 GPU trained for $10M$ steps.

Diff-C has 4 convolutional layers ($conv1$-$conv4$) as shown in Table 10. The final feature dimension is 1024. For all classification tasks and datasets, we train on a NVIDIA RTX A6000 GPU, use a batch size of 16, optimize using AdamW [32], a learning rate of 1×10^{-4}. We train **Diff-C** for 30 epochs with cosine annealing learning rate schedule and set a minimum learning rate η_{min} (5×10^{-5}). We randomly crop and resize input images to 512×512 and augment with random horizontal flip transformations.

4.2. Setup

Datasets and tasks: We interpret and analyze diffusion features of on four image datasets and against competitive baselines on two tasks: (a) **classification** Transfer learning onto Oxford-IIIT Pet [41], FGVC-Aircraft [35], and Caltech-101 [18], (b) **visual reasoning** as in [59], we interleave two visual features with CLIP: DINO [40] and diffusion features. We use the LLaVA-Lightning [3] configuration and MPT-7B-Chat [55] as the base language model. We use CC595k [52] for stage 1 pre-training and LLaVA-Instruct-80K [31] for stage 2 fine-tuning.

Notation: We focus on interpreting the bottleneck and the decoder layers of the UNet, as the information from the encoder is fed into the decoder through the skip connections. As illustrated in Fig. 2, features extracted from a given upsampling (decoder) `block_index` are denoted as `up_ft{block_index}`. `bottleneck` refers to the central block which has the smallest spatial resolution in U-Net. Following the standard convention of reverse denoising of diffusion models [25, 53], $t = 0$ corresponds to the timestep where a final, fully denoised image is achieved and higher values of t represent noisier images, with $t = 1000$ denoting pure noise. For referring to the features from DiT we use `block_index` to denote the transformer block from which we extract the features.

Evaluation metrics: For **Diff-C**, we report top-1 accuracy for all classification tasks. For the visual reasoning task, we evaluate on the LLaVA-Bench (in-the-wild) [31] and MM-Vet benchmarks [67], where model outputs are scored relative to reference answers generated by text-only GPT-4 [4]. For **k-SAE**, to quantify the granularity of semantic information captured in diffusion features, we measure how "pure" the activated k-SAE neurons are. We do this by measuring the average standard deviation in the class labels (σ_{label}) of the top-10 most highly activating images among the top 1000 most highly activating features of the learned k-SAE. We stress that the class labels are not used for training but only to measure activated neurons' purity of the k-SAE. We also visualize images which result in highest activation for a given k-SAE neuron. Furthermore, to further reduce subjectivity, we frame the task of quantifying the granularity of semantic information in diffusion features as a multiple-choice question-answering problem for GPT-4o [3]. We assess the level of semantic detail captured by different diffusion features (prompt details in suppl. material). We note that GPT-4o predictions can be noisy, hence we primarily relied on label purity (σ_{label}) for accurate quantification.

4.3. Information granularity across diffusion layers

In this section, we study how the visual semantic information arranges itself across different layers of a pre-trained

[3] We use LLaVA-Lightning due to compute constraints.

Figure 3. **k-SAE visualizations on Oxford-IIIT Pet** of `bottleneck`, `up_ft1`, and `up_ft2` U-Net layers at $t = 25$. `bottleneck` isolates very coarse patterns of objects positioned similarly with respect to the background. For `up_ft1`, clear class-specific features are observed helping us isolate different fine-grained breeds. `up_ft2` captures more global texture information such as that of grass.

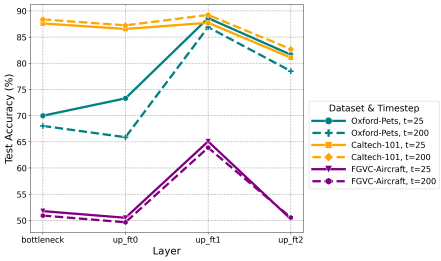

Figure 4. **Top-1 accuracy of different SD 1.5 layer features**. Features from `up_ft1` consistently yield best performance for SD 1.5.

Layer	1. fine-grained	2. Moderately granular	3. Very coarse	4. No patterns
bottleneck	25	52	22	1
up_ft0	33	46	20	1
up_ft1	**47**	41	11	1
up_ft2	28	50	20	2

Table 1. **Accuracy (%)** of GPT-4o predictions among four multiple-choice options, given the top 10 most highly activating images among the top 100 most highly activating neurons of the k-SAE trained on Oxford-IIIT Pet. Of the 4 layers, note how `up_ft1` features are most fine-grained. We note that GPT-4o predictions can be noisy and very sensitive to system prompt.

diffusion network. Specifically, how does the diffusion training objective of minimizing global reconstruction loss impact the visual information granularity across layers? To

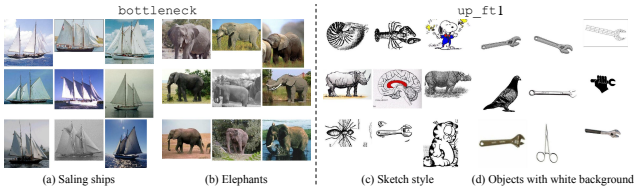

Figure 5. **k-SAE visualizations on Caltech-101** of `bottleneck` and `up_ft1` UNet layers at $t = 25$. Unlike for fine-grained dataset (Fig. 3), `bottleneck` captures class information, likely due to distinct object shapes (sailing ships v/s elephants). `up_ft1` captures more abstract information such as sketches or objects with white background.

this end, we extract diffusion features from `bottleneck`, `up_ft0`, and `up_ft1`, train separate k-SAE and **Diff-C** models, and report evaluation metrics listed in Sec. 4.2.

Fine-grained classification task: From Table 2a, we note that `up_ft1` yields the lowest σ_{label}, indicating that the features corresponding to this layer contain most class-specific information compared to other layers. This is qualitatively corroborated by Fig. 3 where images that k-SAE neurons get most activated by, have very clear class-specific characteristics when using `up_ft1` features compared to `bottleneck` and `up_ft2`. Moreover, the accuracy of GPT-4o predictions (Table 1) suggests that `up_ft1` captures more fine-grained information, whereas other layers tend to capture moderately granular or very coarse features. This finding is also consistent with **Diff-C** results presented in Fig. 4 for Oxford-IIIT Pet and for another fine-grained dataset: FGVC-Aircraft [35]. Note that there is a sharp decline in performance at `up_ft2` layer and beyond, suggesting that `up_ft2` features may be more aligned with the pre-training task objective of pixel reconstruction for image generation, thus are less generalizable for transfer learning. A similar observation was made about the later layers when mechanistically interpreting language models [38].

Does the trend hold for coarse-grained classification task? To deconflate the effect of task-granularity from diffusion feature granularity, we study the diffusion features from Caltech-101 dataset. From Table 2a, it is evident that `bottleneck` features yields a significantly smaller σ_{label}. Also note that the difference between σ_{label} values between layers is quite larger for Caltech-101 compared to Oxford-IIIT Pet. To understand this better, we visualize the highest activated images from different layers. From Fig. 5, we note that those from `bottleneck` are more class-centric and thus "purer" compared to `up_ft1`, which may be capturing more style or texture specific information. We hypothesize that for the task of classifying Caltech-101, coarser shape information is sufficient which is compactly provided by `bottleneck`. This can be clearly seen from Fig. 4, where, compared to Oxford-IIIT Pet, the performance gap between `up_ft1` and `bottleneck` is significantly low. However, for challenging tasks where finer-grained information is required, higher-level layers (`up_ft1`) are more beneficial.

Layer	Oxford-IIIT Pet	Caltech-101	ImageNet
bottleneck	9.48	**9.35**	**25.35**
up_ft0	9.90	15.65	29.10
up_ft1	**8.59**	21.33	34.3
up_ft2	9.67	25.61	36.38

(a) σ_{label} **for different layers:** For Oxford-IIIT Pet, up_ft1 achieves the lowest σ_{label}, whereas bottleneck yields lowest σ_{label} for Caltech-101, indicating the interplay of representation and task granularity.

t	Oxford-IIIT Pet (up_ft1)	Caltech-101 (bottleneck)
0	8.99	11.91
25	**8.59**	9.35
100	8.87	8.72
200	8.94	**8.17**
300	9.01	10.41
500	9.53	16.65

(b) σ_{label} **for different diffusion timesteps:** For Oxford-IIIT Pet up_ft1, $t = 25$ yields the lowest σ_{label}, whereas for Caltech-101 bottleneck, $t = 200$ yields the lowest σ_{label}.

Model	Oxford-IIIT Pet
SD 1.5	**8.59**
SD 2.1	9.67

(c) σ_{label} **for SD 1.5 vs. SD 2.1:** SD 1.5 captures more class-specific information than SD 2.1.

Block	Oxford-IIIT Pet
6	10.18
10	9.44
14	**9.05**
18	9.55
22	9.84

(d) σ_{label} **for different DiT blocks** at $t = 25$. Mid-blocks of DiT yield the lowest σ_{label} than other layers.

Table 2. **Label purity** (σ_{label}) measured by computing the average standard deviation in the class labels (σ_{label}) of the top-10 most highly activating images among the top 1000 most highly activating features of the learned k-SAEs for different diffusion layers, timesteps, models, and architectures on Oxford-IIIT Pet, Caltech-101and ImageNet.

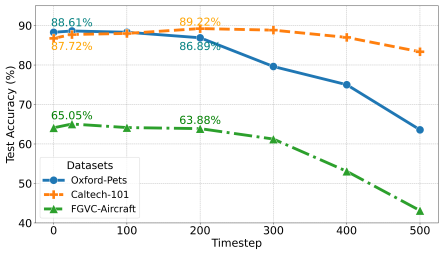

Figure 6. **Top-1 accuracy of up_ft1 at different timesteps:** Earlier timesteps perform better on fine-grained datasets (Oxford-IIIT Pet, FGVC-Aircraft); interim ones on coarse-grained dataset (Caltech-101).

Does the dataset size matter? We conduct the same analysis on ImageNet [16], a relatively coarser-grained dataset. From Table 2a, we observe that even on a larger dataset, the bottleneck layer captures more class-specific information, exhibiting similar behavior as on Caltech-101.

4.4. Information packed across diffusion timesteps

We now examine the interplay between diffusion denoising timesteps and visual semantic information granularity. To this end, we extract diffusion features from up_ft1 at different timesteps $t = \{25, 100, 200, 300, 400, 500\}$, train separate k-SAE and **Diff-C** models. From Table 2b, we observe that $t = 25$ yields the lowest σ_{label}. This is validated both by top activated images shown in Fig. 3 and **Diff-C** performance in Fig. 6. k-SAE neurons are being activated by images with very clear class-specific characteristics when using features extracted at $t = 25$ (more visualizations in suppl. material). This finding is also consistent with **Diff-C** results presented in Fig. 6 for Oxford-IIIT Pet and FGVC-Aircraft. By contrast, from Table 2b, we find that for Caltech-101, features extracted at $t = 200$ yield lowest σ_{label}. This finding is also consistent with **Diff-C** results from Fig. 4. This finding is corroborated in [9, 30]. We hypothesize that the additional noise added at $t = 200$ could be helping in making features more generalizable, but deeper investigation is needed in the future.

4.5. Effect of different models and architectures

Next, we inspect how diffusion models that differ in their underlying architectures, pre-training datasets, attention mechanisms differ in their internal encoding of visual semantic information.

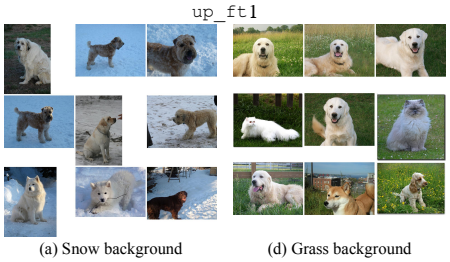

Figure 7. **k-SAE visualizations of up_ft1 of SD 2.1 on Oxford-IIIT Pet** at $t = 25$. Contrary to SD 1.5 (Fig. 3 (c), (d)) where 8 out of 9 images depict the same breed, SD 2.1 features results in 4 in 9 images in (a) as Wheaten Terriers and (b) 5 in 9 images are Great Pyrenees in (b).

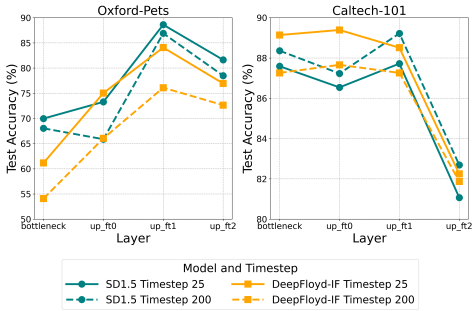

Figure 8. **Stable Diffusion vs DeepFloyd-IF:** The performance dip at up_ft0 is not observed for DeepFloyd-IF across both datasets.

Stable diffusion variants: We study two stable diffusion models (SD 1.5 v/s SD 2.1) which primarily differ in the underlying text encoder and pre-trained datasets. We first extract diffusion features from bottleneck and up_ft1 at $t = \{25, 200\}$ on Oxford-IIIT Pet dataset and train k-SAE and **Diff-C** models. From Table 2c, we note that σ_{label} is lower for SD 1.5 indicating that SD 1.5 captures more object-specific information compared to SD 2.1. This is qualitatively supported by Fig. 3 and Fig. 7 where k-SAE neurons are being activated by images with clearer object-specific information when using SD 1.5 features compared to SD 2.1. This is also consistent with **Diff-C** results in Table 3, where bottleneck features of SD 2.1 are particularly under-performing compared to SD 1.5. Even though there is a sharp performance boost of **13.17**% from using

Model	Params (M)	Layer	t	Test Acc
SD 2.1	900	bottleneck	25	56.80
			200	56.15
		up_ft1	25	84.74
			200	81.77
SD 1.5	893	bottleneck	25	69.97
			200	68.03
		up_ft1	25	**88.61**
			200	86.89
DiT	783	block10	25	87.49
			200	80.89
		block14	25	**88.61**
			200	83.02
DeepFloyd-IF I-900M	900	bottleneck	25	61.16
			200	54.08
		up_ft1	25	84.08
			200	76.09

Table 3. **Top-1 accuracy of different diffusion architectures on Oxford-IIIT Pet.** SD 1.5's up_ft1 and DiT's block 14 perform best overall.

Figure 9. **k-SAE visualizations of DiT blocks on Oxford-IIIT Pet**. Block 14 captures fine-grained information; others capture less distinct features.

up_ft1 for both architectures ($t = 25$), SD 1.5 performs better overall across timesteps. Similar behavior was noted for zero-shot classification in [30] but not well-understood and is a fruitful topic for future research.

Latent v/s pixel space: We next examine how diffusion denoising in the pixel space impacts the learnt visual information differently from those learnt in the latent space. To this end, we compare classification performance of stable diffusion features with those from DeepFloyd-IF [6] which operates directly in the pixel space[4]. Unlike SD 1.5, we note an uptick in the performance of features from up_ft0 layer of DeepFloyd-IF by $\mathbf{1.74}\%$ for Oxford-IIIT Pet, and $\mathbf{2.85}\%$ for Caltech-101 for t=25 as illustrated in Fig. 8. Furthermore, from Table 3, we observe that DeepFloyd-IF's performance is more sensitive to timesteps than Stable Diffusion. For instance, SD 1.5 has a dip of $\mathbf{1.97}\%$ in top-1 accuracy when using bottleneck features at $t = 25$ v/s $t = 200$, while DeepFloyd-IF has a significant drop of $\mathbf{7.08}\%$. Given DeepFloyd-IF operates directly in the pixel space, we think that each denoising step is introducing larger shifts in the underlying semantic structure than in latent space, contributing to these differences.

[4]We acknowledge that despite having similar number of model parameters, both models have different pre-training data, cross-attention connections, and different initial input image resolution.

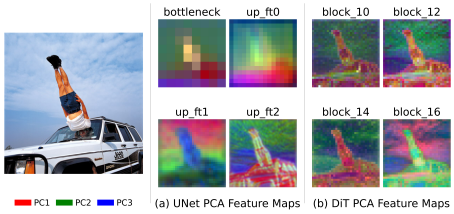

Figure 10. **Visualizing top-3 PCA components of diffusion features** from SD 1.5 and DiT. bottleneck, up_ft0, and up_ft1 of SD 1.5 capture spatially localized information at varied granularity. This property is missing from DiT features across different blocks.

Different diffusion architectures: We also study how semantic information representation varies with the choice of diffusion architecture. To this end, we compare features from U-Net based diffusion model against transformer-based model. Specifically, we extract features from different encoder blocks of DiT [43] and interpret them via both k-SAE and **Diff-C**. From Table 2d, we observe that the middle block of DiT (block14) yields the lowest σ_{label} compared to earlier and later layers. This is qualitatively supported by Fig. 9, where the block14 features contain more class-specific information than other blocks. While with U-Net based features we saw images with spatially related photographic styles emerge (*e.g.*, similar postures or photographic compositions as shown in Fig. 3 (a), (b)), we did not find similar patterns emerge from earlier or later layers of DiT. Though the selected DiT and U-Net based diffusion models have similar number of parameters (Table 3), transformer-based DiT may have less spatial inductive biases compared to the convolutional-based U-Net. Additional visualizations of DiT features in suppl. material.

Inductive biases in diffusion models: To more deeply understand the difference between DiT and SD 1.5 in *how* spatial information is internally encoded, we follow the approach from [60] and apply principle component analysis (PCA) on the diffusion features, and visualize the first three principal components of images from UnRel [44] dataset. From Fig. 10, it is evident that bottleneck features of SD 1.5 capture very coarse spatial information, while up_ft1 capture very clear localized semantic information, even on images where common objects occur out of context. As we go deeper into SD 1.5, layer up_ft2 tends to capture more low-level information (more visualizations in suppl. material). By contrast, DiT's maps exhibit blended colors across all layers, indicating no clear spatially localized information. This property aligns with transformers' tendency to capture more global context by attending to the entire image, and supports k-SAE's interpretations in Fig. 9

LLaVA Vision Encoder	Relative Score
CLIP	56.6
CLIP + DINO-v2	47.0
CLIP + SD 1.5 (up_ft1 at $t = 25$)	**59.9**
CLIP + SD 1.5 (up_ft1 at $t = 200$)	56.8

Table 4. **Performance of multi-modal reasoning task:** SD 1.5's up_ft1 features when integrated with CLIP into LLaVA lead to significant boosts on LLaVA-Bench(In-the-Wild), reflecting alignment with the reference answer generated by text-only GPT-4 responses.

Model	Num Params (M)	Oxford-IIIT Pet	FGVC-Aircraft
Diffusion Classifier (SD-2.0)† [30]	900	87.3*	26.04*
SD-2.0 features [30]	1420	75.9	35.2
Diff-C (up_ft1) - empty	800	88.69	**65.07**
Diff-C (up_ft1) - from_CLIP	800	90.97*	64.98*
CLIP ResNet-50 † [46]	102	85.4	19.3
OpenCLIP (ViT-H/14)† [12]	630	**94.39**	42.75

Table 5. **Top-1 accuracy on Oxford-IIIT Pet and FGVC-Aircraft**. Among models that use diffusion features (top), Diff-C performs best and competes well with CLIP (bottom) † : zero-shot. *: uses text-conditioning.

that there is less spatially-rich information in DiT.

4.6. Performance on visual reasoning

Next, we study the generalizability of diffusion features for visual reasoning by integrating them into the LLaVA [31] framework. Specifically, on the LLaVA-Lightning configuration [31], we extract features from CLIP [46], DINO-v2 [40], and up_ft1 layer from SD 1.5 and pass them independently through separate multi-layer projection layers. We then interleave the projected embeddings as done in [59] and pass them into the language model in LLaVA [31] (implementation details in suppl. material). We use different visual features but keep the language model fixed on the LLaVA-Bench (In-the-Wild) [31] evaluation benchmark. We report the relative scores of the model compared to GPT-4 obtained answers aggregated over three categories (complex reasoning, conversational, and descriptive tasks), more details in suppl. material. From Table 4, it is clear that interleaving CLIP with up_ft1 diffusion features extracted at $t = 25$ improves the relative score by $\mathbf{3.3}\%$. By contrast, interleaving CLIP with DINO-v2 features led to a dip in the performance by $\mathbf{9.6}\%$. Additionally, we evaluate LLaVA on MM-Vet benchmark [67] and find that diffusion features yield a correctness score of 26.4, as assessed by GPT-4o-mini, which is a $+\mathbf{1.1}$ improvement over using only CLIP features. Notably, for the OCR sub task we see a boost of **+2.2** compared to CLIP features. These results suggest that diffusion features, like CLIP, enjoy the benefit of being multi-modal. However, unlike CLIP, diffusion features also encode strong local semantic information (Fig. 10) making them very powerful feature representations.

4.7. State-of-the-art performance

Finally, we compare Diff-C with other models that use diffusion features for representation learning (Table 5 top row) and also with CLIP variants (Table 5 bottom row). In addition to passing an empty prompt which is our default setting, we also experiment with providing a CLIP-inferred prompt during diffusion feature extraction for a fairer comparison with [30]. We note that having access to prompts which presumably can have information about the object in the image puts these models at an unfair advantage, but do this only for comparison. From Table 5, we see that Diff-C performs significantly better than the best reported numbers in [30]: an improvement of $+\mathbf{1.39}\%$ on Oxford-IIIT Pet (88.69) and a huge boost of $+\mathbf{39.03}\%$ for FGVC-Aircraft. "SD-2.0 features" baseline from [30] inputs bottleneck features into ResNet [23] like architecture consuming $520M$ model parameters. On the other hand, Diff-C is a significantly lighter model ($40M$ model parameters) and yet, achieves a huge boost of $+\mathbf{15.07}\%$ for Oxford-IIIT Pet and $+\mathbf{29.87}\%$ for FGVC-Aircraft from using up_ft1 features. This boost clearly illustrates the effectiveness of interpreting the diffusion model states and making an informed selection for achieving the best transfer learning performance on target tasks. It also highlights the benefit of selecting the right visual features over using complex, highly parameterized models. Crucially, the diffusion classifier from [30] takes ≈ 24 sec / sample (using their default settings on Oxford-IIIT Pet) on a single NVIDIA RTX A6000, while Diff-C takes only ≈ 0.13 sec / sample, thereby yielding a 4 orders of magnitude speedup during inference.

Effect of text conditioning: We note that text-conditioning yields mixed results: it leads to performance improvement of Diff-C (rows 3 v/s 4 in Table 5) by $+\mathbf{2.28}\%$ on Oxford-IIIT Pet, but a slight dip of $-\mathbf{0.09}\%$ on FGVC-Aircraft dataset. This detrimental effect of text-conditioning is more pronounced when comparing Diffusion Classifier with SD-2.0 features (rows 1 v/s 2 in Table 5), where the former uses text information, while SD-2.0 features is based purely on visual features. This behavior is not well understood and could be because of the low frequency of occurrence of specific aircraft model names in the natural language captions used for pre-training or a misalignment between pre-training and the domain-specific aircrafts image data.

5. Discussion and Future work

In this work, we present k-sparse auto-encoders as an effective tool to dissect diffusion models of different architectures, across different layers, and inference timesteps. Our qualitative and quantitative analysis shows that the abstraction of visual information oscillates from coarse-grained to fine-grained and then back to coarse-grained as we traverse along the depth of a diffusion model. Fruitful research directions entail effective ways to leverage the interpreted information to design better semantic editing algorithms. We hope that our work will spark more interest on the topic of diffusion model interpretability in the research community.

References

[1] Midjourney. https://www.midjourney.com/home. 3

[2] Introducing gen-3 alpha. 2024. https://runwayml.com/blog/introducing-gen-3-alpha/. 3

[3] gpt 4o. 2024. https://openai.com/index/hello-gpt-4o/. 4, 14

[4] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 4

[5] C. Anil, E. Durmus, M. Sharma, J. Benton, S. Kundu, J. Batson, N. Rimsky, M. Tong, J. Mu, D. Ford, et al. Many-shot jailbreaking. In *NeurIPS*, 2024. 1

[6] DeepFloyd Lab at StabilityAI. DeepFloyd IF: a novel state-of-the-art open-source text-to-image model with a high degree of photorealism and language understanding. https://www.deepfloyd.ai/deepfloyd-if, 2023. Retrieved on 2023-11-08. 7

[7] Y. Ban, R. Wang, T. Zhou, M. Cheng, B. Gong, and C. J. Hsieh. Understanding the impact of negative prompts: When and how do they take effect? *arXiv preprint arXiv:2406.02965*, 2024. 1

[8] F. Bao, S. Nie, K. Xue, Y. Cao, C. Li, H. Su, and J. Zhu. All are worth words: A vit backbone for diffusion models. In *CVPR*, 2023. 2

[9] D. Baranchuk, I. Rubachev, A. Voynov, V. Khrulkov, and A. Babenko. Label-efficient semantic segmentation with diffusion models. *arXiv preprint arXiv:2112.03126*, 2021. 1, 2, 6

[10] T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askell, R. Lasenby, Y. Wu, S. Kravec, N. Schiefer, T. Maxwell, N. Joseph, Z. Hatfield-Dodds, A. Tamkin, K. Nguyen, B. McLean, J. E Burke, T. Hume, S. Carter, T. Henighan, and C. Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html. 2, 3

[11] S. Chen, P. Sun, Y. Song, and P. Luo. Diffusiondet: Diffusion model for object detection. In *ICCV*, 2023. 1, 2

[12] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev. Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023. 8

[13] K. Clark and P. Jaini. Text-to-image diffusion models are zero shot classifiers. In *NeurIPS*, 2024. 2

[14] H. Cunningham, A. Ewart, L. Riggs, R. Huben, and L. Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023. 2

[15] G. Daujotas. Interpreting and steering features in images. 2024. https://www.lesswrong.com/posts/Quqekpvx8BGMMcaem/interpreting-and-steering-features-in-images. 2

[16] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6

[17] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 2

[18] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. In *IEEE TPAMI*, 2006. 4

[19] H. Fry. Towards multimodal interpretability: Learning sparse interpretable features in vision transformers. 2024. https://www.lesswrong.com/posts/bCtbuWraqYTDtuARg/towards-multimodal-interpretability-learning-sparse. 2, 4

[20] L. Gao, Tom D. la T., H. Tillman, G. Goh, R. Troll, A. Radford, I. Sutskever, J. Leike, and J. Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024. 2

[21] R. Girdhar, M. Singh, A. Brown, Q. Duval, S. Azadi, S. S. Rambhatla, A. Shah, X. Yin, D. Parikh, and I. Misra. Emu video: Factorizing text-to-video generation by explicit image conditioning. *arXiv preprint arXiv:2311.10709*, 2023. 3

[22] Q. Guo and D. Yue. Dit-visualization. https://github.com/guoqincode/DiT-Visualization, 2024. Exploring the differences between DiT-based and Unet-based diffusion models in feature aspects using code from diffusers, Plug-and-Play, and PixArt. 2

[23] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 8

[24] X. He, W. Feng, T. J. Fu, V. Jampani, A. Akula, P. Narayana, S. Basu, W. Y. Wang, and X. E. Wang. Discffusion: Discriminative diffusion models as few-shot vision and language learners. *arXiv preprint arXiv:2305.10722*, 2023. 2

[25] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 2, 4

[26] A. Jahanian, X. Puig, Y. Tian, and P. Isola. Generative models as a data source for multiview representation learning. *arXiv preprint arXiv:2106.05258*, 2021. 2

[27] D. P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[28] D. P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[29] M. Kwon, J. Jeong, and Y. Uh. Diffusion models already have a semantic latent space. *arXiv preprint arXiv:2210.10960*, 2022. 2

[30] A. C Li, M. Prabhudesai, S. Duggal, E. Brown, and D. Pathak. Your diffusion model is secretly a zero-shot classifier. In *ICCV*, 2023. 1, 2, 6, 7, 8

[31] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *NeurIPS*, 2024. 4, 8, 11

[32] I. Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 4, 14

[33] G. Luo, L. Dunlap, D. H. Park, A. Holynski, and T. Darrell. Diffusion hyperfeatures: Searching through time and space for semantic correspondence. In *NeurIPS*, 2024. 1, 2

[34] S. Mahajan, T. Rahman, K. M. Yi, and L. Sigal. Prompting hard or hardly prompting: Prompt inversion for text-to-image diffusion models. In *CVPR*, 2024. 2

[35] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 4, 5, 11

[36] A. Makhzani and B. Frey. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013. 2, 3

[37] A. Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72 (2011):1–19, 2011. 3

[38] Nostalgebraist. Interpreting gpt: The logit lens. 2024. https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens. 5

[39] B. A Olshausen and D. J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 1997. 2

[40] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 4, 8, 11

[41] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012. 1, 4, 11

[42] O. Patashnik, D. Garibi, I. Azuri, H. Averbuch-Elor, and D. Cohen-Or. Localizing object-level shape variations with text-to-image diffusion models. In *ICCV*, 2023. 2

[43] W. Peebles and S. Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022. 2, 3, 4, 7

[44] J. Peyre, I. Laptev, C. Schmid, and J. Sivic. Weakly-supervised learning of visual relations. In *ICCV*, 2017. 7, 13

[45] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 3

[46] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 8, 11

[47] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 3

[48] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2, 3, 4

[49] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 4

[50] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L Denton, K. Ghasemipour, Raphael Gontijo L., B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022. 2, 3

[51] L. Sharkey, D. Braun, and B. Millidge. Taking features out of superposition with sparse autoencoders, 2022. *AI Alignment Forum*, 2023. https://www.alignmentforum.org/posts/z6QQJbtpkEAX3Aojj/interim-research-report-taking-features-out-of-superposition. 4

[52] P. Sharma, N. Ding, S. Goodman, and R. Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 4, 11

[53] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 4

[54] V. Surkov, C. Wendler, M. Terekhov, J. Deschenaux, R. West, and C. Gulcehre. Unpacking sdxl turbo: Interpreting text-to-image models with sparse autoencoders. *arXiv preprint arXiv:2410.22366*, 2024. 2

[55] MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms. 2023. www.mosaicml.com/blog/mpt-7b. 4, 11

[56] A. Templeton, T. Conerly, J. Marcus, J. Lindsey, T. Bricken, B. Chen, A. Pearce, C. Citro, E. Ameisen, A. Jones, H. Cunningham, N. L Turner, C. McDougall, M. MacDiarmid, C. D. Freeman, T. R. Sumers, E. Rees, J. Batson, A. Jermyn, S. Carter, C. Olah, and T. Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html. 2

[57] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996. 3

[58] M. Toker, H. Orgad, M. Ventura, D. Arad, and Y. Belinkov. Diffusion lens: Interpreting text encoders in text-to-image pipelines. *arXiv preprint arXiv:2403.05846*, 2024. 2

[59] S. Tong, Z. Liu, Y. Zhai, Y. Ma, Y. LeCun, and S. Xie. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *CVPR*, 2024. 4, 8, 11

[60] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *CVPR*, 2023. 1, 2, 7

[61] A. Voynov, Q. Chu, D. Cohen-Or, and K. Aberman. p+: Extended textual conditioning in text-to-image generation. *arXiv preprint arXiv:2303.09522*, 2023. 2

[62] W. Wang, Q. Sun, F. Zhang, Y. Tang, J. Liu, and X. Wang. Diffusion feedback helps clip see better. *arXiv preprint arXiv:2407.20171*, 2024. 1

[63] W. Wu, Y. Zhao, H. Chen, Y. Gu, R. Zhao, Y. He, H. Zhou, M. Z. Shou, and C. Shen. Datasetdm: Synthesizing data with perception annotations using diffusion models. In *NeurIPS*, 2023. 1

[64] J. Xu, S. Liu, A. Vahdat, W. Byeon, X. Wang, and S. De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *CVPR*, 2023. 1, 2, 11

[65] X. Yang and X. Wang. Diffusion model as representation learner. In *ICCV*, 2023. 2

[66] J. Ye, N. Wang, and X. Wang. Featurenerf: Learning generalizable nerfs by distilling foundation models. In *ICCV*, 2023. 2

[67] W. Yu, Z. Yang, L. Li, J. Wang, K. Lin, Z. Liu, X. Wang, and L. Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023. 4, 8

[68] W. Zhao, Y. Rao, Z. Liu, B. Liu, J. Zhou, and J. Lu. Unleashing text-to-image diffusion models for visual perception. In *ICCV*, 2023. 1

Revelio: Interpreting and leveraging semantic information in diffusion models

Supplementary Material

A. Text Conditioning in Diffusion Models

Following the findings from [64], we report the performance of **Diff-C** in two text conditioning scenarios: i) empty prompt and ii) a meaningful prompt, e.g., "a photo of a $\{class_name\}$, a type of pet", with the $class_name$ first inferred through a zero-shot classification with CLIP. The motivation behind reporting both scores is to provide a comprehensive understanding of how text conditioning affects visual features at each layer. We report the classification performance with and without CLIP-inferred captions in Tables 6, 7, and 8. We note that passing specific class information inferred from CLIP generally helps across all three datasets, layers, and timesteps. To further understand how specific the captions should be, we experiment by passing a generic prompt, e.g., "a photo of a pet" during the diffusion process. As shown in Table 9 for `up_ft1` layer, on Oxford-IIIT Pet [41], compared to the base setting of passing in an empty prompt, using a generic prompt leads to a performance drop by 3.14%. This indicates that the specificity of the text being used to condition directly impacts feature representation quality, where more targeted prompts align better with class-relevant features, thereby improving model accuracy. Consequently, using precise text conditioning can lead to considerable gains in performance, particularly in distinguishing nuanced categories. However, this may not always be the case as described in Sec. 4.7, where for FGVC-Aircraft [35] conditioning with the class names led to a dip in classification performance.

B. Layer-wise PCA Analysis of Feature Maps

Figures 11 and 12 provides more evidence to the findings in Sec. 4.5. by highlighting differences in how SD 1.5 and DiT encode spatial information. In SD 1.5, the feature maps reveal well-defined spatial structures, with consistent colors and textures that correspond to specific regions in the image. By contrast, the feature maps of DiT display blended patterns, suggesting a stronger focus on capturing global context rather than emphasizing distinct spatial details.

C. Additional Details on the Visual Reasoning Task

Hyper-parameters: We adopt the same hyperparameters used in the the LLaVA-Lightning [31] configuration across all experiments. We use MPT-7B-Chat [55] as the language model, and CLIP ViT-L/14 [46], DINOv2 ViT-L/14 [40], SD 1.5 as the vision encoders. We show the training hyperparameters in Table 12. All experiments were conducted using a maximum of 4 NVIDIA RTX A6000 GPUs.

Pre-training datasets: Following LLaVA-Lightning [31], we use CC595k [52] for stage 1 pre-training, to align the visual encoder with the language model to establish a shared vision-language representation, by tuning the adapter. For stage 2 fine-tuning we use LLaVA-Instruct-80K [31] to fine-tune the model to enhance instruction-following capabilities.

Adapter settings: For experiments involving CLIP and DINOv2 features, we use the standard 2 layer MLP projector to align visual tokens with language tokens [31]. To obtain tokenized representations from the feature maps obtained from SD 1.5, we first add a 2 layer convolutional block and transform the feature map into pseudo-tokenized representations that match the token embedding dimensions of CLIP and DINOv2. These pseudo-tokenized representations are then passed into the 2 layer MLP projector for alignment.

Interleaving diffusion features with CLIP for visual reasoning tasks: For the experiments reported in Sec. 4.6, we first gradually reduce the spatial dimension of `up_ft1` from $1280 \times 32 \times 32$ to 256×1024 to match the token dimensions of CLIP vision embeddings. Next, we process these embeddings through two separate multi-layer projection layers resulting in projected embeddings of shape 256×4096. Finally, we interleave the projected token embeddings as done in [59] before passing them into LLaVA [31].

Performance: Table 11 compares the performance of different vision encoders in LLaVA, including CLIP (Table 11a), CLIP+DINOv2 (Table 11b), and CLIP+Diffusion at timesteps $t = 25$ (Table 11c) and $t = 200$ (Table 11d). The evaluation is conducted on the LLaVA-Bench (in-the-wild) [31] benchmark. The benchmark evaluates models across four categories: overall performance ('all'), complex reasoning ('LLaVA Bench complex'), conversational tasks ('LLaVA Bench conversational'), and descriptive tasks ('LLaVA Bench detail').

For the 'detail' category, CLIP+Diffusion at $t = 25$ achieves the highest relative score of **56.2**, outperforming both CLIP (50.4) and CLIP+DINOv2 (37.7). This demonstrates that the interleaved diffusion and CLIP features effectively capture fine-grained visual details. In the

Timestep (t)	bottleneck (empty / from_CLIP)	up_ft0 (empty / from_CLIP)	up_ft1 (empty / from_CLIP)	up_ft2 (empty / from_CLIP)
0	52.27 / 52.74	48.79 / 49.06	64.09 / 62.826	50.55 / 49.15
25	51.76 / 54.88	50.49 / 51.19	65.07 / 63.69	50.25 / 49.21
100	51.07 / 55.12	49.48 / 51.10	64.15 / 64.98	51.37 / 50.53
200	50.91 / 52.51	49.63 / 49.99	63.88 / 63.13	50.53 / 50.55

Table 6. **Top-1 accuracy** at different timesteps and layers for fine-grained task (FGVC-Aircraft).

Timestep (t)	bottleneck (empty / from_CLIP)	up_ft0 (empty / from_CLIP)	up_ft1 (empty / from_CLIP)	up_ft2 (empty / from_CLIP)
0	68.79 / 84.33	66.99 / 79.50	88.28 / 90.11	77.79 / 80.67
25	69.97 / 85.25	73.29 / 84.17	88.61 / 90.68	81.63 / 85.28
100	69.53 / 85.88	67.07 / 81.33	88.29 / 90.97	78.82 / 84.36
200	68.03 / 86.43	65.87 / 81.79	86.89 / 90.32	78.49 / 84.79

Table 7. **Top-1 accuracy** at different timesteps and layers for fine-grained task (Oxford-IIIT Pet).

Timestep (t)	bottleneck (empty / from_CLIP)	up_ft0 (empty / from_CLIP)	up_ft1 (empty / from_CLIP)	up_ft2 (empty / from_CLIP)
0	85.83 / 92.13	85.75 / 89.68	86.75 / 88.18	79.11 / 80.28
25	87.59 / 91.32	86.54 / 90.81	87.72 / 91.08	81.07 / 82.69
100	88.28 / 92.41	88.18 / 90.66	87.99 / 92.05	82.02 / 84.73
200	88.36 / 91.65	87.23 / 90.73	89.22 / 92.26	82.69 / 85.63

Table 8. **Top-1 accuracy** at different timesteps for coarse-grained task (Caltech-101).

Prompt Type	Timestep	
	25	200
Empty Prompt	88.61	86.89
from_CLIP	2.34 ↑	3.94 ↑
generic	3.14 ↓	3.13 ↓

Table 9. **Performance vs Text Conditioning on Oxford-IIIT Pet using `up_ft1`:** Using a generic prompt ("A photo of a pet") leads to a dip in classification performance compared to using an empty prompt. By contrast, using a targeted caption ("A photo of a $\{class_name\}$, a type of pet") leads to a boost in performance.

Layer	Output Shape	Description
conv1	$[B, 1024, H, W]$	conv2D, 3×3
conv2	$[B, 1024, H/2, W/2]$	conv2D, 3×3, stride 2
conv3	$[B, 1024, H/4, W/4]$	conv2D, 3×3, stride 2
conv4	$[B, 1024, H/8, W/8]$	conv2D, 3×3, stride 2
GAP	$[B, 1024, 1, 1]$	global average pooling
FC	$[B, \texttt{NUM_CLASSES}]$	flatten + FC layer

Table 10. Architecture of **Diff-C** ($40M$ params).

'complex' category, CLIP+Diffusion at $t = 200$ achieves the highest relative score of **70.5**, surpassing CLIP (68.4). At $t = 25$, CLIP+Diffusion scores 67.9 indicating that the coarser-grained features extracted at higher timesteps ($t = 200$) seem more effective for this specific task that requires broader contextual understanding. Next, for the 'conversational' category, CLIP+Diffusion at $t = 25$ achieves a relative score of **51.3**, outperforming both CLIP (43.9) and CLIP+DINOv2 (35.6). The interleaving of diffusion and CLIP features significantly enhances the model's ability to handle visually grounded conversational tasks effectively.

Finally, we report the overall performance under the 'all' category and note that CLIP+Diffusion achieves a superior performance with a score of **59.9** at $t = 25$, outperforming CLIP's standalone score of 56.6. This reinforces the power of the visual representations learnt from the diffusion process in achieving top-performance on diverse vision-language tasks.

D. Additional k-SAE Visualizations

DiT vs U-Net: In this section, we provide additional visualizations of k-SAE features. As shown in Fig. 13 (b), (e), `Block 14` of DiT captures more class-specific information than other blocks which is qualitatively corroborated in Table 2d. However, compared to SD 1.5, DiT captures less distinct class information, as seen in the snow background in Fig. 13 (h). Moreover, the spatially related photographic styles observed in Sec. 4.5 do not emerge in DiT. We hypothesize that the transformer-based relies less on inductive bias information compared to UNet-based SD 1.5, as discussed in Sec. 4.5.

Later timesteps: Figure 14 presents k-SAE visualization at $t = 500$ for SD 1.5. Compared to $t = 25$, features at $t = 500$ focus more on low-level information, such as texture and low-light, which is qualitatively corroborated in

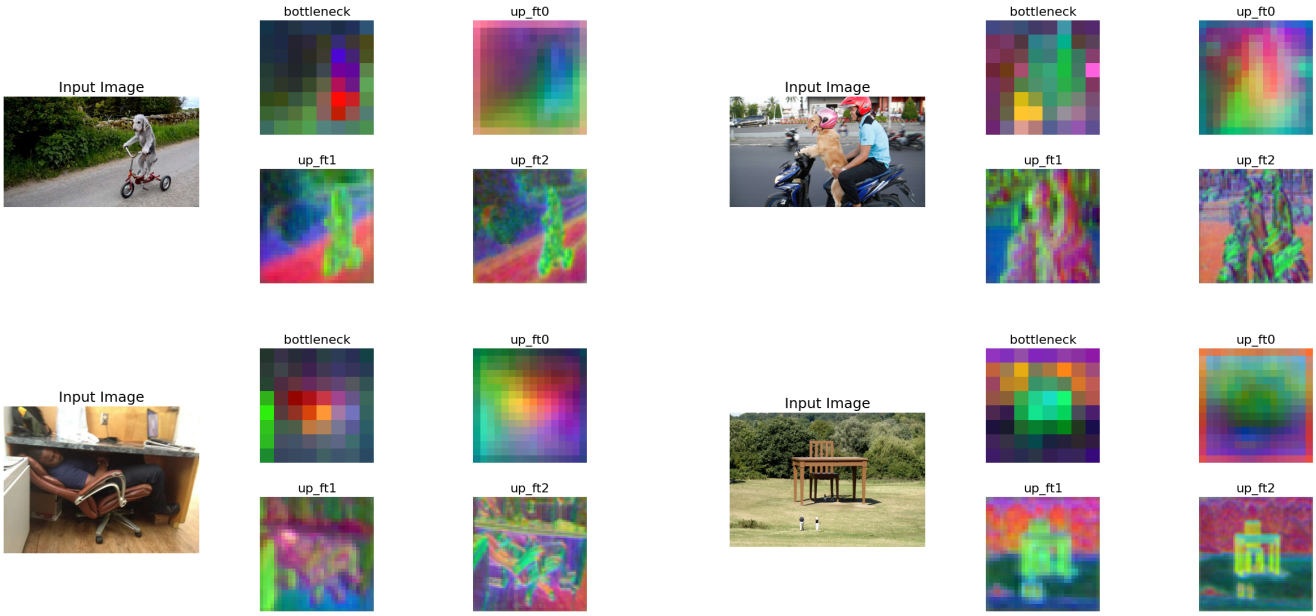

Figure 11. **PCA Feature Maps SD-1.5 on images** from UnRel [44] - Consistency of colors and textures (at up_ft1, up_ft2) suggests that the model preserves local details and spatial relationships

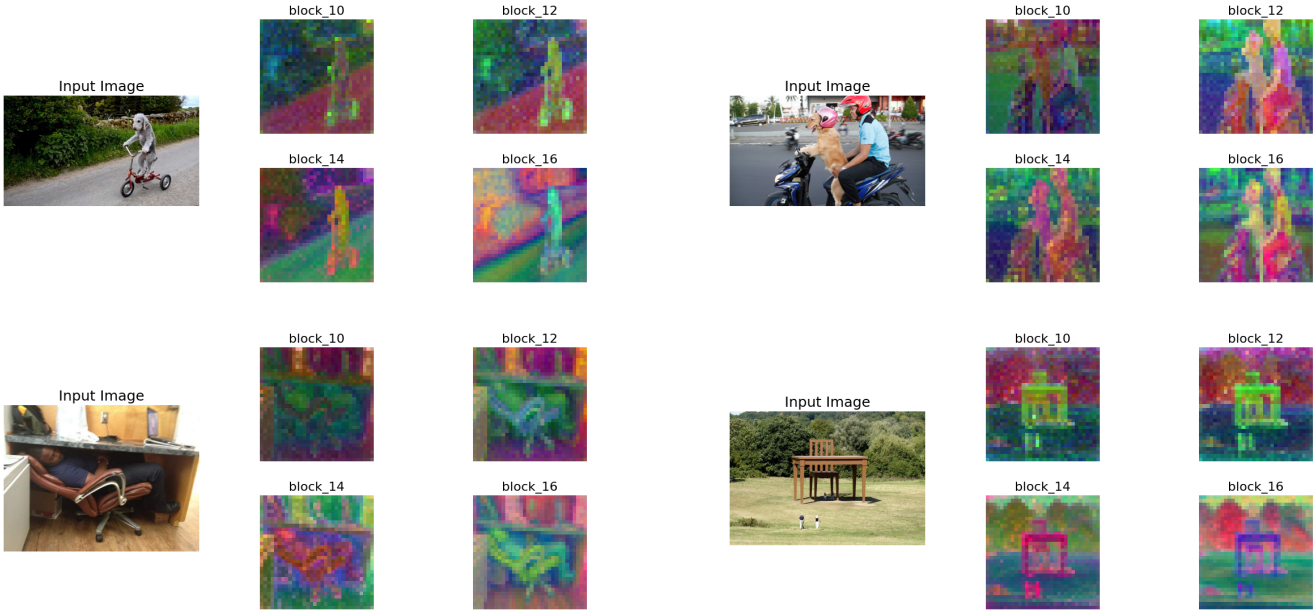

Figure 12. **PCA Feature Maps DiT on images from UnRe**l [44] - The blending of colors suggests that the model encodes global relationships while maintaining a holistic representation of spatial structures, rather than isolating precise local details.

Table 2b. We hypothesize that as the diffusion timestep increases, so does the added noise, rendering the features less useful for transfer learning, consistent with our observations in Sec. 4.4.

E. Additional k-SAE Experiments

Effect of image resolutions: To assess the robustness of our method across varying input resolutions, we conduct additional experiments examining the effect of image resolution. As shown in Table 13, using different image resolutions exhibits a similar trend in terms of σ_{label}, with smaller resolutions resulting in slightly reduced variance across dif-

Category	Relative Score	GPT-4 Score	LLaVA Score
All	56.6	82.7	46.8
LLaVA Bench complex	68.4	80.4	55.0
LLaVA Bench conversational	43.9	87.1	38.2
LLaVA Bench detail	50.4	82.0	41.3

(a) CLIP LLaVA

Category	Relative Score	GPT-4 Score	LLaVA Score
All	47.0	84.8	39.8
LLaVA Bench complex	59.9	81.1	48.6
LLaVA Bench conversational	35.6	94.1	33.5
LLaVA Bench detail	37.7	81.3	30.7

(b) CLIP+DINOv2 LLaVA

Category	Relative Score	GPT-4 Score	LLaVA Score
All	**59.9**	83.2	49.8
LLaVA Bench complex	67.9	80.0	54.3
LLaVA Bench conversational	**51.3**	90.6	46.5
LLaVA Bench detail	**56.2**	80.7	45.3

(c) CLIP+Diffusion ($t = 25$) LLaVA

Category	Relative Score	GPT-4 Score	LLaVA Score
All	56.8	83.7	47.5
LLaVA Bench complex	**70.5**	80.0	56.4
LLaVA Bench conversational	45.6	87.7	40.0
LLaVA Bench detail	45.7	86.0	39.3

(d) CLIP+Diffusion ($t = 200$) LLaVA

Table 11. **Performance on the multi-modal reasoning task for various LLaVA configurations**. The integration of Diffusion features with CLIP improves performance across all tasks, with notable gains in the 'detail' and 'conversational' categories.

Hyperparameter	Stage	
	Stage 1	Stage 2
batch size	128	128
learning rate (lr)	2e-3	2e-5
lr schedule decay	cosine	cosine
lr warmup ratio	0.03	0.03
weight decay	0	0
epoch	1	1
optimizer	AdamW [32]	
deepspeed stage	2	3

Table 12. Hyperparameters for LLaVA-Lightning (default setting)

ferent DiT blocks on Oxford-IIIT Pet.

F. Additional Details of Evaluation

In this section, we provide additional details on how we quantify the granularity of semantic information in diffusion features through a multiple-choice question-answering task, as discussed in Sec. 4.3. Using GPT-4o [3], we evaluate the level of semantic detail captured by different diffusion features. Table 14 presents the prompt used to query the model for this evaluation. Specifically, we assess the

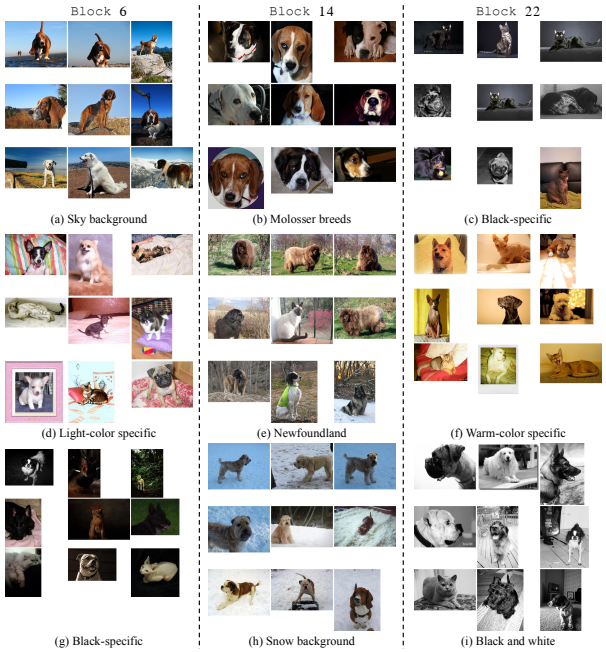

Figure 13. **k-SAE visualizations of the blocks on Oxford-IIIT Pet** at $t = 25$. `Block 14` mainly captures class-specific information, while other blocks focus more on less distinct features.

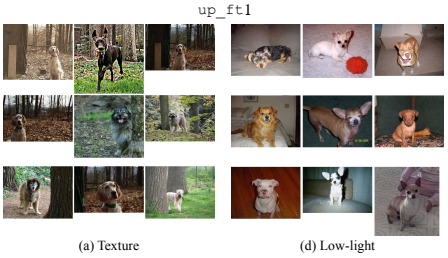

Figure 14. **k-SAE visualizations on Oxford-IIIT Pet** of `up_ft1` UNet layer at $t = 500$. In contrast to the earlier timestep (Fig 3), $t = 500$ appears to focus more on low-level features.

Block	Oxford-IIIT Pet	
	(512)	(256)
6	10.18	10.36
10	9.44	10.16
14	**9.05**	**10.06**
18	9.55	10.11
22	9.84	10.13

Table 13. **Label purity** (σ_{label}) measured by computing the average standard deviation of the class labels of the top-10 most highly activating images among the top 1000 most highly activating features of the learned k-SAEs for different DiT blocks with different resolutions on Oxford-IIIT Pet. Lower is better.

model's predictions based on the top 10 most highly activating images among the top 100 most highly activating neurons of the learned k-SAE.

```
Prompt: Each set of images captures
    different types of patterns:
1. Class-specific information (e.g., fine-
    grained details, animals of the same
    breed).
2. Moderately granular features (e.g.,
    similar-looking animals irrespective
    of their position).
3. Very coarse information (e.g.,
    foreground objects similarly placed
    relative to the background).
4. Could not detect patterns (e.g., noisy
    or no specific patterns).
Select only one number (1, 2, 3, or 4)
    that best describes the shared pattern
**Respond with just the number and nothing
     else.**
```

Table 14. Input prompt for GPT-4o based evaluation.