SAE++: Cascaded Sparse Autoencoders Learn Multi-Level Visual Concepts in Multimodal LLMs

Yusong Zhao[1,*,†,‡] **Hengyi Wang**[2,*] **Tanuja Ganu**[3] **Akshay Nambi**[3] **Hao Wang**[1]

Abstract

Multimodal Large Language Models (MLLMs) have demonstrated strong performance on vision-language tasks, yet their internal visual representations remain difficult to interpret. Sparse Autoencoders (SAEs) provide a scalable way to decompose dense model activations into sparse, interpretable features. However, existing SAE architectures primarily recover flat feature dictionaries and are less suited for explicit multi-level concept organization. In this paper, we introduce a cascaded sparse autoencoder architecture, dubbed SAE++, for learning hierarchical visual concepts in MLLMs. Rather than nesting or stacking SAE sparse activation codes, SAE++ trains a second-level SAE directly on the decoder weights of the first-level SAE, treating learned low-level feature directions as inputs for higher-level abstraction. This design enables SAE++ to learn "concepts of concepts" while avoiding drawbacks from the shared-prefix coupling of nesting, Matryoshka-style hierarchies and the bottlenecks of naively stacked SAEs. Experiments across Qwen3-VL, Gemma-3, and LLaVA on multiple visual datasets show that SAE++ improves interpretability in terms of hierarchical concept coherence over state-of-the-art SAE baselines. Results on concept steering further demonstrate that the learned concept groups support effective group-level interventions in MLLM outputs. Code is available at https://github.com/Wang-ML-Lab/sae-plus-plus.

1 Introduction

Multimodal Large Language Models (MLLMs) such as LLaVA [1], Qwen-VL [2], and Gemma [3] have demonstrated exceptional capabilities in processing visual inputs beyond the purely textual scope of traditional LLMs. They bridge the gap between advanced visual perception and the reasoning abilities of Large Language Models, achieving state-of-the-art performance in complex tasks ranging from visual question answering and embodied agency to general-purpose personal assistance [4, 5]. Despite their rapid deployment in commercial and everyday applications, most of these models remain "black boxes", often exhibiting unpredictable and hazardous behaviors. These include severe hallucinations, such as confidently describing non-existent objects [6, 7], and vulnerability to visual jailbreak attacks that bypass safety alignment [8, 9]. These risks highlight the need for interpretable representations and predictions [10], particularly the concepts encoded inside MLLMs.

However, the mechanistic interpretation of MLLMs presents challenges that exceed those of traditional deep learning models. A primary obstacle is the intrinsic polysemy of deep neural networks, where a single neuron may activate for disparate concepts [11, 12]. Moreover, the high-dimensional nature of internal MLLM layers further exacerbates this issue. Given the open-ended behavior of MLLMs, disentangling these features through manual annotation is infeasible.

To address these challenges, Sparse Autoencoders (SAEs) [13] have emerged as a scalable, unsupervised methodology. By projecting dense model activations into an overcomplete, sparse latent

[1]University of Illinois Urbana-Champaign. [2]Rutgers University. [3]Microsoft Research. [*]Co-first authors. [†]Project lead. [‡]Correspondence to: Yusong Zhao <yusongz@illinois.edu>.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

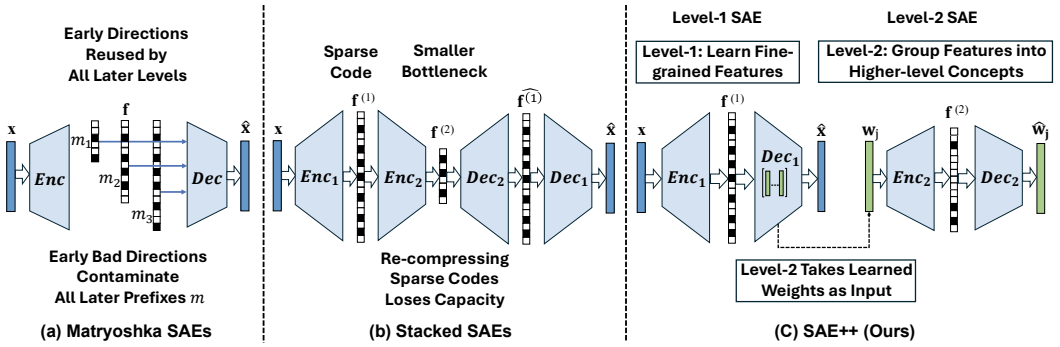


Figure 1: Overview of hierarchical design choices for SAEs. **(a) Matryoshka SAEs** construct the concept hierarchy through a single nested prefix chain, with early latent directions globally reused across later levels. **(b) Stacked SAEs** learn the concept hierarchy by re-compressing sparse sample codes through a smaller bottleneck, which introduces an additional capacity constraint. **(c) Our SAE++** takes a different route: the Level-1 SAE learns low-level features, and the Level-2 SAE is trained directly on the learned Level-1 decoder weights, enabling higher-level abstraction over concepts rather than overly shared prefixes or re-compressed sample codes.

space, SAEs explicitly attempt to decompose superimposed features into distinct, monosemantic directions. SAEs have been successfully used to discover diverse features in both large language models [14] and vision-language models [15, 16, 17, 18, 19]. Recent multimodal SAE studies further show that sparse features can help analyze vision-language alignment, shared embedding spaces, and data quality in MLLMs. Despite these advances, learning explicit multi-level visual concept hierarchies from MLLMs remains challenging.

Recent work attempts to address this issue using Matryoshka SAEs inspired by Matryoshka representation learning [20, 21, 16], discovering hierarchical features in models such as CLIP [22] and Gemma-2-2b. However, these methods implement hierarchy through nested prefixes of a single dictionary, so early directions are reused by downstream prefixes, creating a shared-prefix coupling that can limit explicit multi-level concept organization; error in high-level concepts can propagate to low-level concepts (theoretical analysis in Sec. 4.2). Another natural approach is to stack multiple SAE layers. However, stacked SAEs compress sparse activation codes through a smaller high-level bottleneck, which can lose information needed for reconstruction and thus hinder the learning of lower-level concepts (theoretical analysis in Sec. 4.1).

Fig. 1 shows an overview comparing different designs. To address this challenge, this paper makes a systematic attempt to *explore an alternative beyond the nested architecture in Matryoshka SAEs and the stacked SAE architecture*. We propose, to the best of our knowledge, the first cascaded SAE architecture, dubbed SAE++. Rather than feeding the activations of one SAE layer into another, we jointly train two SAEs: a high-level SAE and a low-level SAE. The low-level (Level-1) SAE learns low-level concepts directly from MLLM activations. **Our key insight** is that the MLLM activations, Level-1 decoder weight columns, and Level-2 decoder weight columns all lie in the same d-dimensional activation space. This enables a recursive "representatives of representatives" construction: the Level-1 decoder atoms represent MLLM activations, while the Level-2 SAE treats these learned concept directions as its inputs and learns higher-level representatives of them, i.e., "concepts of concepts." Our contributions are:

- We propose the first cascaded SAE architecture, dubbed SAE++, that enables the learning of hierarchical concepts from MLLMs.
- Theoretical analysis shows that naively stacking SAE layers can suffer from a sparse bottleneck failure mode, while Matryoshka-style nested prefixes can amplify local semantic errors through shared-prefix reuse.
- Empirical results across Qwen3-VL, Gemma-3, and LLaVA on multiple visual datasets demonstrate improved hierarchical concept interpretability. Results on activation steering show that the learned concept groups enable effective interventions in MLLM outputs.

2 Related Work

Multimodal LLMs. Compared to traditional text-only LLMs, Multimodal LLMs (MLLMs) perceive and reason across modalities such as images, video, audio, and text. Seminal works such as CLIP [22]

aligned visual and textual representations for zero-shot classification, Flamingo [23] introduced interleaved visual-textual modeling, and BLIP-2 [24] adapted frozen LLMs to visual tasks using lightweight adapters. Modern multimodal systems include closed-source models such as the GPT series [25], Gemini series [26, 27, 28], and Claude [29], as well as open-source models such as LLaVA [1], Qwen-VL [30, 31, 32], DeepSeek-VL [33], Pixtral [34], Gemma [3, 35], and Llama 3.2 Vision [36]. While these models show strong performance, it remains unclear how they represent and organize visual concepts internally; this is the focus of our paper.

Sparse Autoencoders. Recently, Sparse Autoencoders (SAEs) have emerged as a powerful tool to enhance the mechanistic interpretability of large models. Generally, SAEs [13, 14] employ an overcomplete set of basis vectors and a sparsity penalty (typically ℓ_1) to decompose the dense, polysemantic activations of a neural network into linear combinations of interpretable, monosemantic feature directions. Existing variants include foundational ℓ_1-based SAEs [13, 37], Top-K sparsity [38], Gated SAEs [39], and JumpReLU SAEs [40]. These architectures improve sparse feature learning through mechanisms such as hard sparsity, learnable gating, and discontinuous activations, but they are primarily designed for flat feature dictionaries. Although SAEs facilitate the interpretability of large models, adapting the aforementioned structures to discover *hierarchical* (i.e. multi-level) concepts within multimodal LLMs presents unique challenges.

Matryoshka SAEs [21, 16] learn hierarchical concepts by training multiple nested dictionaries of increasing size, with smaller dictionaries encouraged to reconstruct inputs independently, so earlier prefixes learn more general concepts and later prefixes refine them with more specific features. However, because these hierarchies are implemented through shared nested prefixes, early directions are reused by downstream prefixes, creating a shared-prefix coupling. Consequently, error in high-level concepts can propagate to low-level concepts (see theoretical analysis in Sec. 4.2). In contrast, our proposed SAE++ learns high-level concepts by training a second SAE directly on the decoder weight columns of the low-level SAE, enabling explicit abstraction over learned visual concepts.

Interpreting Multimodal Models with SAEs. SAEs have been extensively used to interpret large models. Early work studied text-only LLMs, showing that SAEs can decompose polysemantic activations into more interpretable features [13, 14]. Later work scaled SAEs to larger language models such as GPT-2/GPT-4 [38], Claude 3 Sonnet [41], and Gemma 2 [42]. More recently, concept-based methods have been applied to vision and multimodal models [43, 44]. Prior work trains SAEs on CLIP representations to isolate visual concepts [16, 45]. Other studies analyze multimodal alignment and shared embedding spaces: SAE-V interprets MLLM cross-modal features and uses them for data filtering and alignment improvement [17]; VL-SAE maps visual and textual representations into a unified concept set for interpreting and enhancing vision-language alignment [18]; and SAEs trained on VLM embedding spaces reveal sparse linear structures shaped by modality and cross-modal semantic bridges [19]. These works demonstrate the utility of SAEs for multimodal interpretability and alignment. In contrast, our work studies explicit *multi-level visual concept hierarchies* in modern MLLMs, where high-level concepts organize coherent groups of low-level concepts.

3 Methodology

3.1 Preliminary: Sparse Autoencoders (SAEs)

Sparse Autoencoders (SAEs) implement a form of sparse dictionary learning designed to decompose the dense, polysemantic activations of large models into interpretable components. Given an input embedding vector $\mathbf{x} \in \mathbb{R}^d$ from a specific layer of a (multimodal) LLM, the goal is to learn an overcomplete dictionary of features parameterized by an encoder $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{n \times d}$ and a decoder $\mathbf{W}_{\text{dec}} \in \mathbb{R}^{d \times n}$ (where $n \gg d$). Crucially, the architecture typically employs a shared geometric bias $\mathbf{b} \in \mathbb{R}^d$, which is subtracted from the input to center the signal during encoding and added back to the output during decoding.

The encoding process $g : \mathbb{R}^d \to \mathbb{R}^n$ projects the centered input into a sparse latent space, while the decoding process $h : \mathbb{R}^n \to \mathbb{R}^d$ maps the activations back to the input data manifold:

$$\mathbf{f} = g(\mathbf{x}) = \sigma(\mathbf{W}_{\text{enc}}(\mathbf{x} - \mathbf{b})), \tag{1}$$

$$\widehat{\mathbf{x}} = h(\mathbf{f}) = \mathbf{W}_{\text{dec}}\mathbf{f} + \mathbf{b}, \tag{2}$$

where $\sigma(\cdot)$ is a non-linear activation function (e.g., ReLU). The model parameters are learned by minimizing a joint objective $\mathcal{L}$ composed of a reconstruction term $\mathcal{R}$ and a sparsity penalty $\mathcal{S}$:

$$\mathcal{L}(\mathbf{x}) = \underbrace{||\mathbf{x} - \widehat{\mathbf{x}}||_2^2}_{\text{reconstruction } \mathcal{R}(\mathbf{x})} + \underbrace{\lambda \mathcal{S}\big(g(\mathbf{x})\big)}_{\text{sparsity penalty}}, \tag{3}$$

where $\mathcal{S}(\cdot)$ is usually an ℓ_1 loss, i.e., $\|\cdot\|_1$, to encourage sparsity. While the standard instantiation relies on ReLU activation and an ℓ_1 sparsity penalty, this framework readily accommodates alternative architectures. For instance, TopK SAEs [38] substitute the soft ℓ_1 penalty with a hard sparsity constraint by modifying $\sigma(\cdot)$ to select only the top-k activations.

Similarly, Matryoshka SAEs [21] organize the latent space into nested index subsets $\mathcal{I}_i = \{1, \ldots, m_i\}$ with $m_1 < \cdots < m_L$, enforcing a joint objective $\sum_{i=1}^{L} \|\mathbf{x} - (\mathbf{W}_{\text{dec}}^{(:,\mathcal{I}_i)}\mathbf{f}_{\mathcal{I}_i} + \mathbf{b})\|_2^2$ so that smaller prefixes are encouraged to reconstruct the input independently.

3.2 From SAE to SAE++

Existing SAEs usually treat dictionary atoms as independent vectors, failing to capture the hierarchical correlations inherent in multimodal data (e.g., lower-level concepts "barrel" and "drum" together form the higher-level concept of "cylinder").

To address this, we propose a new SAE architecture, SAE++, which is a two-level sparse dictionary learning system trained end-to-end. This framework implements hierarchical feature abstraction, systematically organizing atomic concepts (features) from the low-level SAE into high-level semantic structures. Below we introduce SAE++ in detail. Note that while we describe SAE++ as a *two-level* model, our formulation *can be naturally generalized to more than two levels, as shown in Appendix C*.

Let $\mathbf{x} \in \mathbb{R}^d$ denote an activation vector from a fixed layer of a multimodal LLM. Our SAE++ consists of two hierarchical levels trained jointly to capture both fine-grained atomic features and their high-level semantic structures. Fig. 1(c) shows an overview of SAE++.

Level-1 (Low-Level) SAE. We employ a standard SAE to decompose the input activations $\mathbf{x}$. This Level-1 SAE has an encoder g_1 and a decoder h_1, which produce a sparse code $\mathbf{f}^{(1)} \in \mathbb{R}^{n_1}$ and a reconstruction $\widehat{\mathbf{x}}$:

$$\mathbf{f}^{(1)} = g_1(\mathbf{x}) = \sigma(\mathbf{W}_{\text{enc}}^{(1)}(\mathbf{x} - \mathbf{b}^{(1)})), \tag{4}$$

$$\widehat{\mathbf{x}} = h_1(\mathbf{f}^{(1)}) = \mathbf{W}_{\text{dec}}^{(1)}\mathbf{f}^{(\mathbf{1})} + \mathbf{b}^{(1)}. \tag{5}$$

Importantly, here the decoder weight matrix $\mathbf{W}_{\text{dec}}^{(1)}$ contains the atomic concept (feature) directions learned by the model. We denote the columns of this decoder by:

$$\mathbf{W}_{\text{dec}}^{(1)} = [\mathbf{w}_1, \ldots, \mathbf{w}_{n_1}], \quad \mathbf{w}_j \in \mathbb{R}^d, \tag{6}$$

where each $\mathbf{w}_j$ represents a specific concept direction in the original activation space. For each input LLM activation embedding $\mathbf{x}$, we can define the Level-1 loss as follows:

$$\mathcal{L}_1(\mathbf{x}, \boldsymbol{\theta}_1) = ||\mathbf{x} - \widehat{\mathbf{x}}||_2^2 + \lambda_1 \mathcal{S}\big(g_1(\mathbf{x})\big),$$

where $\mathcal{S}(\cdot)$ is usually an ℓ_1 loss, i.e., $\|\cdot\|_1$, to encourage sparsity. $\boldsymbol{\theta}_1 = \{\mathbf{W}_{enc}^{(1)}, \mathbf{W}_{dec}^{(1)}, \mathbf{b}^{(1)}\}$ denotes parameters in the Level-1 SAE.

Level-2 (High-Level) SAE. The core innovation of our SAE++ lies in the structural "clustering" performed by the Level-2 (high-level) SAE. Rather than processing features $\mathbf{x}$ or $\mathbf{f}^{(1)}$, our Level-2 SAE operates directly on the atomic concept directions, i.e., the weight columns $\{\mathbf{w}_j\}_{j=1}^{n_1}$ of the Level-1 decoder $\mathbf{W}_{\text{dec}}^{(1)}$ (Eqn. (6)). It uses an encoder-decoder pair to map each Level-1 atom $\mathbf{w}_j$ from the original feature space $\mathbb{R}^d$ to a higher-level latent space $\mathbb{R}^{n_2}$ and reconstruct it back to $\mathbb{R}^d$:

$$\mathbf{f}_j^{(2)} = g_2(\mathbf{w}_j) = \sigma(\mathbf{W}_{\text{enc}}^{(2)}(\mathbf{w}_j - \mathbf{b}^{(2)})), \tag{7}$$

$$\widehat{\mathbf{w}}_j = h_2(\mathbf{f}_j^{(2)}) = \mathbf{W}_{\text{dec}}^{(2)}\mathbf{f}_j^{(2)} + \mathbf{b}^{(2)}. \tag{8}$$

Our key insight is that the MLLM activations and decoder columns at both levels lie in the same space $\mathbb{R}^d$. Level 2 therefore learns representatives of the Level-1 concept directions, i.e., "concepts of concepts" in the original activation space. This Level-2 (high-level) SAE is learned with the objective:

$$\mathcal{L}_2(\mathbf{W}_{\text{dec}}^{(1)}, \boldsymbol{\theta}_2) = \frac{1}{n_1} \sum_{j=1}^{n_1} \Big(\|\mathbf{w}_j - h_2(g_2(\mathbf{w}_j))\|_2^2 + \lambda_2 \mathcal{S}(g_2(\mathbf{w}_j)) \Big), \tag{9}$$

where $\mathcal{S}(\cdot)$ is usually an ℓ_1 loss, i.e., $\|\cdot\|_1$, to encourage sparsity. $\boldsymbol{\theta}_2 = \{\mathbf{W}_{enc}^{(2)}, \mathbf{W}_{dec}^{(2)}, \mathbf{b}^{(2)}\}$ denotes parameters in the Level-2 SAE.

Joint Optimization. The final objective combines the losses of the Level-1 SAE ($\mathcal{L}_1$) and the Level-2 SAE ($\mathcal{L}_2$):

$$\mathcal{L}_{\text{final}}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \mathbb{E}_{\mathbf{x}}[\mathcal{L}_1(\mathbf{x}, \boldsymbol{\theta}_1)] + \alpha \mathcal{L}_2(\mathbf{W}_{\text{dec}}^{(1)}, \boldsymbol{\theta}_2). \tag{10}$$

Here, α is a hyperparameter that balances low-level concept learning and the high-level concept abstraction. Note that $\mathbf{W}_{\text{dec}}^{(1)}$ is part of the Level-1 SAE's parameters $\boldsymbol{\theta}_1$. Empirical training diagnostics for the joint objective are provided in Appendix D.5.

Dynamic Masking of Dead Latents. For computational efficiency, we apply the Level-2 SAE only to active Level-1 decoder atoms (weight columns) $\mathbf{w}_j$ in the current mini-batch. At training step t, given mini-batch $\mathcal{B}$, we define $\mathcal{A}_t = \{\mathbf{w}_j \mid \sum_{\mathbf{x}\in\mathcal{B}} \mathbf{1}(|[g_1(\mathbf{x})]_j| > \epsilon_0) \geq 1\}$, where $\mathbf{w}_j$ is the j-th Level-1 decoder atom, $[g_1(\mathbf{x})]_j$ is its Level-1 activation on input $\mathbf{x}$, and $\epsilon_0 > 0$ is a small threshold. We compute the Level-2 loss only over $\mathcal{A}_t$. This masking affects only which Level-1 atoms are included in the Level-2 objective, not the architecture or final objective (more details in Appendix E.2).

3.3 Discussion and Insight: SAE++ Learns Concepts of Concepts

Level-1 Decoder's Weight Columns $\mathbf{w}_j$ Are Representative Points in the MLLM Activation Space. Our key observation is that the Level-1 decoder weight columns and the original MLLM activations lie in the same space and have the same dimensionality d. Specifically, given an MLLM activation $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{W}_{dec}^{(1)} \in \mathbb{R}^{d\times n_1}$ has d rows and n_1 columns, and each column $\mathbf{w}_j \in \mathbb{R}^d$ also has d dimensions. Eqn. (5), $h_1(f^{(1)}) = \mathbf{W}_{dec}^{(1)} f^{(1)} + \mathbf{b}^{(1)}$, reconstructs each MLLM activation as a sparse linear combination of these columns. Thus, although $\mathbf{W}_{dec}^{(1)}$ is formally a parameter matrix, its columns are learned d-dimensional "representative points" (i.e., concepts) of the d-dimensional MLLM activation embeddings, rather than arbitrary model weights.

Level-2 Decoder's Weight Columns Are "Representatives of Representatives". This directly motivates our Level-2 SAE. If the Level-1 SAE learns "representative" concepts from MLLM activations, Level 2 learns "representatives of these representatives", i.e., "concepts of concepts." Specifically, it maps each $\mathbf{w}_j \in \mathbb{R}^d$ to a sparse code $f_j^{(2)} = g_2(\mathbf{w}_j)$ in $\mathbb{R}^{n_2}$ and reconstructs it using $h_2(f_j^{(2)}) = \mathbf{W}_{dec}^{(2)} f_j^{(2)} + \mathbf{b}^{(2)}$. Therefore, Level-1 concepts such as "Truck" and "Jeep" can both be mapped to the same higher-level (Level-2) factor corresponding to "Vehicle". Note that the Level-2 decoder weight columns and the Level-1 decoder weight columns are also in the same space and have the same dimensionality d.

To summarize, (1) the Level-2 decoder weight columns, (2) the Level-1 decoder weight columns, and (3) the original MLLM activations lie in the same space and have the same dimensionality d. Therefore, our SAE++'s cascaded architecture ensures that

- the Level-1 decoder weight columns are representative points (i.e., Level-1 concepts) of the MLLM activations and that
- the Level-2 decoder weight columns are representative points (i.e., "representatives of these representatives" or "concepts of concepts") of the Level-1 decoder weight columns.

4 Theoretical Analysis

In this section, we provide theoretical analysis on why two natural hierarchical SAE designs, Stacked SAEs and Matryoshka SAEs, may fall short in learning multi-level concepts. Specifically, we show that these two designs may fail for different reasons: stacked SAEs face a sparse-code bottleneck

mismatch, while Matryoshka SAEs can reuse early semantic errors across downstream prefixes. *The purpose of this section is not to prove that these baselines are always ineffective, but to isolate structural failure modes that our SAE++ avoids by construction.* These results identify the structural effects of sparse-code bottlenecks and nested-prefix reuse, providing a theoretical rationale for SAE++'s decoder-atom cascade. **All proofs are in Appendix B.**

4.1 Why Stacked SAEs Fail

Failure of Stacked SAEs. Consider a stacked SAE with the architecture (similar to Fig. 1(b))

$$d \to n \to m \to n \to d, \tag{11}$$

where the first SAE (the encoder-decoder pair $d \to n$ and $n \to d$) maps an activation to a sparse code

$$\mathbf{z} \in \mathcal{S}_{n,k_1}(B) := \{\mathbf{z} \in \mathbb{R}^n : \|\mathbf{z}\|_0 \le k_1,\ \|\mathbf{z}\|_2 \le B\}.$$

The second SAE uses $g : \mathbb{R}^n \to \mathbb{R}^m$ and $h : \mathbb{R}^m \to \mathbb{R}^n$ to reconstruct $\mathbf{z}$. If its bottleneck is sparser,

$$g(\mathbf{z}) \in \mathcal{S}_{m,k_2}(R) := \{\mathbf{u} \in \mathbb{R}^m : \|\mathbf{u}\|_0 \le k_2,\ \|\mathbf{u}\|_2 \le R\}, \qquad k_2 < k_1,$$

then the second SAE must preserve a k_1-sparse code space using a k_2-sparse bottleneck. This is exactly the setting one might hope would learn higher-level abstractions: a lower-dimensional and sparser code should represent more compressed concepts. The theorem below shows that this intuition conflicts with uniform reconstruction.

Theorem 4.1 (**Failure of Stacked SAEs with a Sparser Bottleneck**). *Assume uniform reconstruction*

$$\sup_{\mathbf{z} \in \mathcal{S}_{n,k_1}(B)} \|h(g(\mathbf{z})) - \mathbf{z}\|_2 \le \varepsilon,$$

where h is L_h-Lipschitz, and assume $g(\mathbf{z}) \in \mathcal{S}_{m,k_2}(R)$ with $k_2 < k_1$. If $\varepsilon \le B/8$ and the support-packing capacity inequality in Appendix B.1, Eq. (14)*, holds, no such encoder-decoder pair can uniformly reconstruct $\mathcal{S}_{n,k_1}(B)$. For the empirically used $k_1 = 6$, $k_2 = 2$, this inequality holds for sufficiently large n when $m < n$ and RL_h/B remains bounded.*

Theorem 4.1 shows why naive stacking is not an effective architecture: it compresses the combinatorial space of sparse sample codes through a smaller sparse bottleneck. The issue is not caused by a particular optimizer or activation function; it follows from the geometry of sparse code spaces.

SAE++Avoids Similar Failure. In contrast, SAE++ avoids this issue because its Level-2 SAE is trained on Level-1 decoder atoms $\mathbf{w}_j$,

$$\mathcal{L}_2 = \frac{1}{n_1} \sum_{j=1}^{n_1} \left(\|\mathbf{w}_j - h_2(g_2(\mathbf{w}_j))\|_2^2 + \lambda_2 \mathcal{S}(g_2(\mathbf{w}_j)) \right),$$

where $\mathbf{w}_j \in \mathbb{R}^d$ is a Level-1 decoder weight column. Thus, the second level abstracts over concept directions rather than sparse sample codes (more details and theoretical analysis in Appendix B).

4.2 Shared-Prefix Error Amplification in Matryoshka SAEs

Shared-Prefix Reuse of Matryoshka SAEs. Matryoshka SAEs build hierarchy by ordering decoder columns $\mathbf{d}_1, \mathbf{d}_2, \ldots$, each a feature **direction** in activation space (like $\mathbf{w}_1, \mathbf{w}_2, \ldots$ in SAE++). The t-th prefix uses the first m_t directions, and all later prefixes reuse them. Hence, early directions are shared across levels. If such a direction captures reconstruction-relevant but *semantically irrelevant* variation, its error propagates across multiple levels. We refer to such directions as **nuisance directions**.

Formalizing Prefix Reuse. Matryoshka SAEs optimize nested index sets $\mathcal{I}_t = \{1, \ldots, m_t\},\ m_1 < \cdots < m_L$, with the objective function $\sum_{t=1}^{L} \left\| \mathbf{x} - \left(\mathbf{W}_{\text{dec}}^{(:,\mathcal{I}_t)} \mathbf{f}_{\mathcal{I}_t} + \mathbf{b} \right) \right\|_2^2$, where $\mathbf{W}_{\text{dec}}^{(:,\mathcal{I}_t)}$ and $\mathbf{f}_{\mathcal{I}_t}$ are subsets of $\mathbf{W}_{\text{dec}}$ and $\mathbf{f}$ indexed by $\mathcal{I}_t$. Because prefix t can only use the first m_t columns, any column at position p is reused in all prefixes with $m_t \ge p$. We define the reuse count $C_{\text{MSAE}}(p) := |\{t : m_t \ge p\}|$, which quantifies how often a direction is reused; earlier columns have larger reuse.

From Prefix Reuse to Semantic Mismatch. We analyze semantics via an orthonormal linear surrogate. Let $D = [\mathbf{d}_1, \ldots, \mathbf{d}_{m_L}],\ U_t(D) = \text{span}\{\mathbf{d}_1, \ldots, \mathbf{d}_{m_t}\}$. With projector P_U, prefix t

reconstructs by projection, giving the idealized reconstruction loss $\mathbb{E}_{\mathbf{x}}\|\mathbf{x} - P_{U_t(D)}\mathbf{x}\|_2^2$. This linear assumption is used only for the shared-prefix analysis; details are in Appendix B.2.

To measure semantic correctness, we compare $U_t(D)$ with an ideal semantic subspace $\mathcal{M}_t$ (see Appendix B.2 for the derivation of the projector-distance form):

$$\mathcal{E}_t^{\mathrm{MSAE}}(D) = \|P_{U_t(D)} - P_{\mathcal{M}_t}\|_F^2, \quad \mathcal{E}_{\mathrm{MSAE}}^{\mathrm{tot}}(D) = \sum\nolimits_{t=1}^{L} \mathcal{E}_t^{\mathrm{MSAE}}(D).$$

Thus, mismatch reflects deviation from intended semantic structure.

Normalized Semantic Mismatch. Because early directions appear in many prefixes, their errors are counted repeatedly in $\mathcal{E}_{\mathrm{MSAE}}^{\mathrm{tot}}$. To isolate local effects, we define a normalized error counting each nuisance direction once.

Let nuisance directions be $\mathbf{d}_{p_i} = \mathbf{n}_i$, $i = 1, \ldots, q$, each orthogonal to all $\mathcal{M}_t$. For prefix t with $m_t \geq p_i$, define a repaired subspace $\widetilde{U}_t^{(-i)}(D)$ replacing $\mathbf{n}_i$. The local error reduction is

$$e_{i,t}^{\mathrm{MSAE}} = \mathcal{E}_t^{\mathrm{MSAE}}(D) - \|P_{\widetilde{U}_t^{(-i)}(D)} - P_{\mathcal{M}_t}\|_F^2.$$

Averaging over reused prefixes yields

$$\overline{\mathcal{E}}_{\mathrm{MSAE}} = \sum\nolimits_{i=1}^{q} \frac{1}{C_{\mathrm{MSAE}}(p_i)} \sum\nolimits_{t:m_t \geq p_i} e_{i,t}^{\mathrm{MSAE}}.$$

This removes duplication effects and enables *fair comparison between Matryoshka SAEs and SAE++*.

Semantic Mismatch for SAE++. In contrast, SAE++ does not reuse directions across levels. Each Level-1 atom $\mathbf{w}_j$ has a single parent assignment. Let $c(j)$ be its ideal parent, with subspace $\mathcal{P}_{c(j)}$, and $\widehat{\mathcal{P}}_j$ the assigned one. The local error is

$$e_j^{\mathrm{SAE++}} = \|P_{\widehat{\mathcal{P}}_j} - P_{\mathcal{P}_{c(j)}}\|_F^2,$$

and for a set $\mathcal{B}$,

$$\overline{\mathcal{E}}_{\mathrm{SAE++}}(\mathcal{B}) = \sum\nolimits_{j \in \mathcal{B}} e_j^{\mathrm{SAE++}}.$$

Since each error is counted once in SAE++, no reuse normalization is needed.

Theorem 4.2 (Prefix Reuse Amplifies Matryoshka Semantic Error). *Assume q mutually orthonormal nuisance directions $\mathbf{n}_1, \ldots, \mathbf{n}_q$ are orthogonal to every target semantic subspace $\mathcal{M}_t$, and occupy positions $p_1, \ldots, p_q$ in D. Then $\mathcal{E}_{\mathrm{MSAE}}^{\mathrm{tot}}(D) \geq 2\sum_{i=1}^{q} C_{\mathrm{MSAE}}(p_i)$. Moreover, for any $\mathcal{B}$ with $|\mathcal{B}| \leq q$, we have*

$$\overline{\mathcal{E}}_{\mathrm{SAE++}}(\mathcal{B}) \leq 2q \leq \overline{\mathcal{E}}_{\mathrm{MSAE}}. \tag{12}$$

SAE++ Keeps Semantic Errors Local. Theorem 4.2 shows that the same number of local semantic errors can lead to a larger error in Matryoshka SAEs than in SAE++. In Matryoshka SAEs, an erroneous early decoder direction is reused by multiple prefixes, so its error is propagated to lower-level (children) concepts. In SAE++, each erroneous atom assignment is counted once and does not propagate to lower-level concepts. Thus, SAE++ avoids shared-prefix amplification and yields *smaller or identical semantic error* compared to Matryoshka SAEs under the same local error budget.

5 Experiments

We evaluate SAE++ against multiple SAE baselines on three MLLM models and four visual datasets.

5.1 Experimental Setup

MLLMs and Datasets. We evaluate our method using `Qwen3-VL-4B-Instruct`, `Gemma-3-4B-IT`, and `LLaVA-1.5-13B`. Activations are extracted from Qwen3-VL visual block 23, Gemma-3 vision tower layer 26, and LLaVA language backbone layer 39, respectively, using the prompt "Describe the image content accurately." We use Color [46], ImageNet [47], iNaturalist [48], and COCO [49]. Dataset sampling and implementation details are provided in Appendix E.1.

Table 1: **HMS**$_{\text{mean}}$ for different methods. More HMS results are in Appendix F. We mark the best results in **bold** and the second best with underline.

MLLM	Data	BTK	TopK	ReLU	P-Ann.	GSAE	JReLU	MSAE	Stack.	SAE++
Qwen3	Color	0.421	0.122	0.214	0.545	0.655	0.147	0.978	0.601	**0.983**
	ImageNet	0.256	0.155	0.562	0.639	0.561	0.076	0.584	0.332	**0.778**
	COCO	0.127	0.151	0.328	0.126	0.281	0.121	0.461	0.144	**0.980**
	iNat.	0.318	0.325	0.193	0.416	0.261	0.102	0.588	0.231	**0.741**
Gemma-3	Color	0.667	0.445	0.222	0.660	0.566	0.160	0.985	0.710	**0.993**
	ImageNet	0.161	0.134	0.080	0.150	0.256	0.157	0.687	0.488	**0.775**
	COCO	0.280	0.053	0.466	0.160	0.666	0.167	0.702	0.412	**0.891**
	iNat.	0.503	0.230	0.089	0.148	0.225	0.415	0.869	0.460	**0.913**
LLaVA	Color	0.331	0.242	0.317	0.336	0.432	0.434	0.962	0.577	**0.971**
	ImageNet	0.191	0.246	0.110	0.493	0.195	0.410	0.747	0.230	**0.896**
	COCO	0.397	0.396	0.321	0.334	0.090	0.237	0.811	0.284	**0.941**
	iNat.	0.333	0.316	0.364	0.266	0.363	0.258	0.759	0.260	**0.780**

Table 2: **HMS**$_{\text{med}}$ for different methods. More HMS results are in Appendix F. We mark the best results in **bold** and the second best with underline.

MLLM	Data	BTK	TopK	ReLU	P-Ann.	GSAE	JReLU	MSAE	Stack.	SAE++
Qwen3	Color	0.420	0.108	0.213	0.545	0.520	0.127	0.999	0.628	**0.999**
	ImageNet	0.141	0.139	0.604	0.798	0.561	0.068	0.615	0.337	**1.000**
	COCO	0.122	0.144	0.126	0.126	0.192	0.117	0.460	0.147	**1.000**
	iNaturalist	0.244	0.265	0.200	0.140	0.193	0.100	0.598	0.225	**0.974**
Gemma-3	Color	0.641	0.389	0.224	0.658	0.612	0.130	0.985	0.661	**0.994**
	ImageNet	0.159	0.135	0.080	0.090	0.088	0.087	0.753	0.499	**0.838**
	COCO	0.280	0.053	0.466	0.160	0.828	0.174	0.725	0.508	**1.000**
	iNaturalist	0.503	0.230	0.089	0.148	0.096	0.144	0.886	0.456	**0.961**
LLaVA	Color	0.322	0.240	0.248	0.325	0.353	0.309	0.938	0.679	**0.948**
	ImageNet	0.175	0.218	0.081	0.498	0.198	0.276	0.779	0.199	**0.965**
	COCO	0.330	0.338	0.254	0.271	0.090	0.207	0.865	0.203	**0.997**
	iNaturalist	0.298	0.287	0.312	0.242	0.293	0.212	0.786	0.229	**0.987**

Baselines. We compare with BatchTopK (**BTK**) [50], **TopK** SAE [38], **ReLU** SAE [13], P-Annealing (**P-Ann.**) [37], Gated SAE (**GSAE**) [39], JumpReLU (**JReLU**) [40], Matryoshka SAE (**MSAE**) [21, 16], and Stacked SAE (**Stack.**). More baseline details are in Appendix E.4.

Metrics. We evaluate multi-level semantic coherence with Hierarchical Mono-Semanticity (HMS), adapted from the mono-semanticity score [45]. For each Level-1 concept j, we compute a semantic vector $\mathbf{R}_j$ as an activation-weighted average of fixed-backbone image embeddings, using the normalized activations described in Appendix E.5. Let $\pi(j)$ denote the Level-2 parent of Level-1 concept j, and let $\mathcal{C}_k = \{j : \pi(j) = k\}$ be the children of Level-2 concept k. Similar to [45], we define

$$\mathrm{HMS}(k) = \frac{2}{m(m-1)} \sum_{\substack{u,v \in \mathcal{C}_k \\ u<v}} \frac{\mathbf{R}_u^\top \mathbf{R}_v}{\|\mathbf{R}_u\|_2 \|\mathbf{R}_v\|_2}, \qquad m = |\mathcal{C}_k|.$$

HMS measures whether Level-1 concepts under the same Level-2 parent are semantically coherent. Cluster sizes, atom coverage, a shared parent-assignment control, and matched-sparsity reconstruction results are provided in Appendix D. In the main paper, we report $\mathrm{HMS}_{\text{mean}} = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \mathrm{HMS}(k)$ and $\mathrm{HMS}_{\text{med}} = \mathrm{median}_{k \in \mathcal{K}} \mathrm{HMS}(k)$ over eligible parents $\mathcal{K} = \{k : |\mathcal{C}_k| \geq 2\}$, Results for more metrics, $\mathrm{HMS}_{\min}$ and $\mathrm{HMS}_{\max}$, are in Appendix E.5 and F. For concept steering, we use four LLM judges to evaluate whether the target concept appears after insertion or is removed after suppression. More details are provided in Appendix D.3.

5.2 Results

Quantitative Results. Table 1 and Table 2 show $\mathrm{HMS}_{\text{mean}}$ and $\mathrm{HMS}_{\text{med}}$ across three MLLMs and four datasets. SAE++ achieves the best $\mathrm{HMS}_{\text{mean}}$ and $\mathrm{HMS}_{\text{med}}$ in all settings, with especially large gains on COCO and ImageNet, where learning high-level concepts is more challenging. Matryoshka SAE is usually the strongest baseline, but it consistently trails SAE++, suggesting that explicit concept

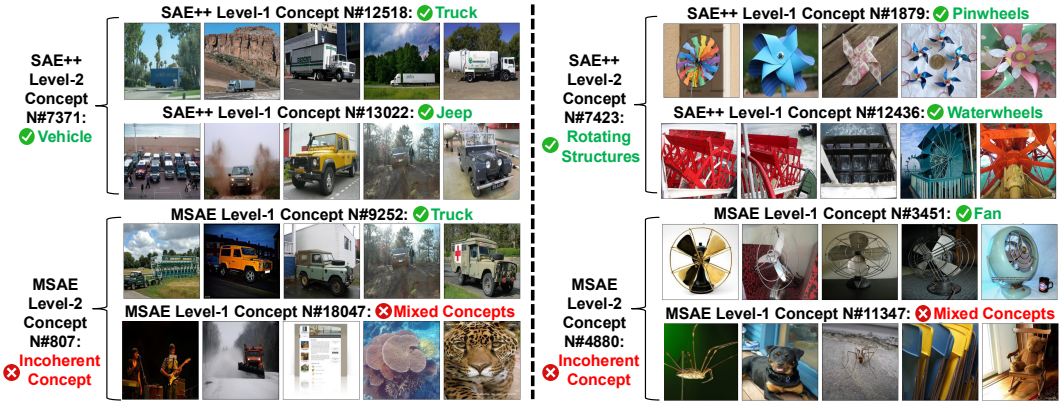


Figure 2: Qualitative examples of multi-level (Level-1 and Level-2) concepts discovered by SAE++ and Matryoshka SAEs (MSAEs). **Left:** SAE++ groups visually coherent Level-1 concepts, *Truck* and *Jeep*, into the same Level-2 concept, *Vehicle*. In contrast, the matched MSAE concepts are less semantically consistent and often mix weaker or unrelated visual patterns. **Right:** SAE++ groups visually coherent Level-1 concepts, *Pinwheels* and *Waterwheels*, into the same Level-2 concept, *Rotating Structures*, while the two Level-1 concepts under the same MSAE Level-2 concept are less semantically consistent. Additional qualitative results are provided in Appendix G.

Table 3: Concept steering success rates (%) on `Qwen3-VL-4B-Instruct` under four LLM judges. We report success rates on 1,000 COCO evaluations from ten Level-2 concepts; each judge scores the same generated outputs. Judge Avg. is the unweighted mean across the four judges. Higher is better.

	Insertion (%)					Suppression (%)				
Judge	**NS**	**Rand.**	**MSAE**	**Single**	**SAE++**	**NS**	**Rand.**	**MSAE**	**Single**	**SAE++**
Gemini-2.5-Flash	10.6	10.8	40.7	58.4	**67.8**	0.0	12.4	25.2	35.6	**40.5**
Gemini-3-Flash	9.8	12.1	41.3	57.5	**70.4**	0.0	14.5	21.9	34.2	**43.6**
Nemotron-30B	17.1	16.4	40.7	62.7	**73.3**	0.0	13.2	24.5	36.3	**40.6**
Opus-4.8	4.7	5.2	41.0	60.9	**72.6**	0.0	9.6	22.6	30.3	**35.7**
Judge Avg.	10.6	11.1	40.9	59.9	**71.0**	0.0	12.4	23.6	34.1	**40.1**

abstraction yields more coherent high-level concepts than shared-prefix hierarchies. Complete HMS results, including HMS_{min}, HMS_{med}, HMS_{max} , and HMS_{mean}, are provided in Appendix F.

Qualitative Results. Fig. 2 visualizes representative multi-level concepts discovered by SAE++ and Matryoshka SAE. SAE++ groups fine-grained but related Level-1 concepts under coherent Level-2 abstractions, such as *Truck*/*Jeep* under a *Vehicle* concept and *Pinwheels*/*Waterwheels* under a *Rotating Structure* concept. Matched Matryoshka concepts are related but often mix less coherent visual patterns. These examples support the HMS results: SAE++ preserves visually specific Level-1 concepts while organizing them into coherent high-level groups. Additional examples are provided in Appendix G.

Concept Steering. Beyond concept discovery, we evaluate whether SAE++ supports group-level interventions. On COCO test images, we randomly select 10 Level-2 concepts and 100 images per concept, yielding 1,000 evaluations. We steer all SAE Level-1 concepts under the same Level-2 concept and feed the reconstructed activations back into the MLLM residual stream. The steering scale $\mathbf{u}_s$ is computed from the mean activation of the top-10 training images for each cluster; we use $+3\mathbf{u}_s$ for concept insertion and $-3\mathbf{u}_s$ for concept suppression. We compare against no steering (**NS**), random concepts (**Rand.**), matched Matryoshka SAE clusters (**MSAE**), and best individual SAE++ Level-1 concept (**Single**). We evaluate the same generated outputs using four LLM judges. As shown in Table 3, SAE++ achieves the highest insertion and suppression success rates under every judge, with four-judge averages of 71.0% and 40.1%, respectively.

Fig. 3 shows a case study on the "*Table*" concept. We compare unsteered generation, matched Matryoshka SAE steering, the best individual SAE++ Level-1 neuron, and full SAE++ Level-2

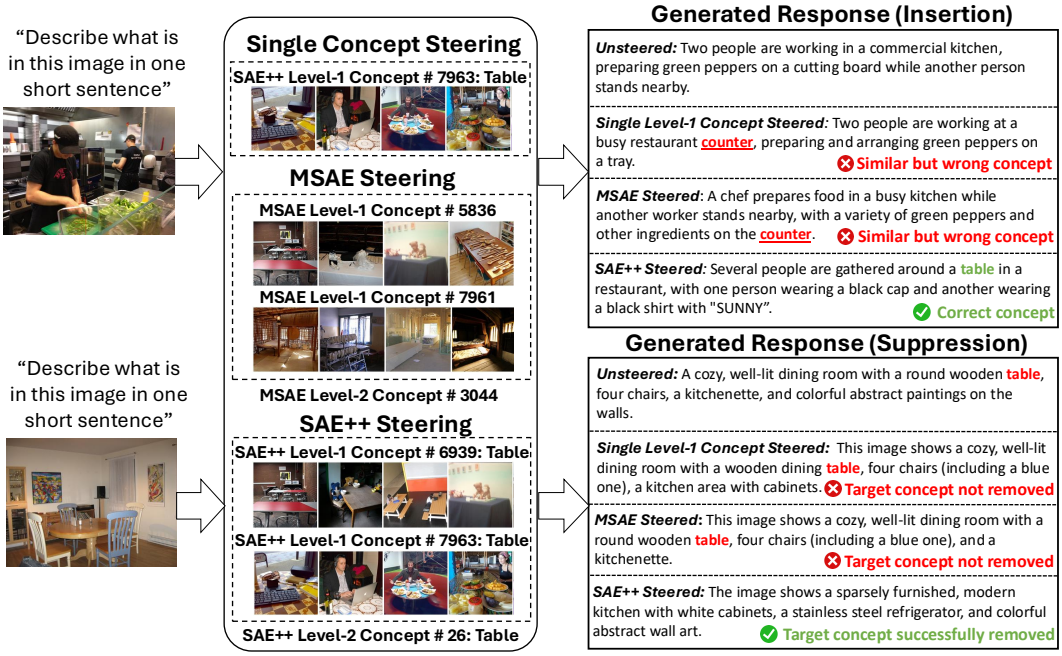


Figure 3: Concept steering on `Qwen3-VL-4B-Instruct`. We compare unsteered generation, matched Matryoshka SAE steering, the best individual SAE++ Level-1 neuron, and full SAE++ Level-2 clustered steering. In both insertion and suppression, SAE++'s steering most effectively controls the generated response in terms of the "*Table*" concept.

Table 4: Ablation studies on ImageNet with `Qwen3-VL-4B-Instruct`. Results on more HMS metrics are available in Appendix F.

Metric	Partial SG	Bilateral SG	Stacked	HAC	SC	SAE++ (Full)
$\mathbf{HMS}_{\text{med}}$	0.8280	0.7012	0.3367	<u>0.8587</u>	0.6906	**0.9999**
$\mathbf{HMS}_{\text{mean}}$	<u>0.7099</u>	0.6333	0.3322	0.7025	0.6490	**0.7776**

clustered steering. In both insertion and suppression, SAE++'s steering most effectively controls the generated response in terms of the "*Table*" concept. More details are provided in Appendix E.6.

Ablation Studies. Table 4 evaluates which components are responsible for the gains of SAE++: (1) Restricting gradient flow between the two SAE levels hurts performance: both partial and bilateral stop-gradient variants (**Partial SG** and **Bilateral SG**, details in Appendix E.3) lead to lower HMS scores compared with full joint training (**SAE++ (Full)**), indicating that the Level-2 objective should shape the Level-1 atoms rather than merely cluster a fixed dictionary. (2) The Stacked SAE (**Stacked**) performs substantially worse, supporting our theoretical argument that naively re-compressing sparse activation codes is poorly suited for multi-level concept learning. (3) Replacing the Level-2 SAE with post-hoc clustering on Level-1 decoder weight columns $\mathbf{w}_j$, including Hierarchical Agglomerative Clustering (**HAC**) and Spectral Clustering (**SC**), also underperforms our full SAE++. This suggests that the learned Level-2 sparse abstraction is more effective than post-hoc clustering.

6 Conclusion

We introduced SAE++, a cascaded SAE framework for hierarchical concept discovery in MLLMs. By training a Level-2 SAE on Level-1 decoder atoms, SAE++ learns abstractions over concept directions while avoiding the bottleneck and shared-prefix limitations of stacked and Matryoshka-style designs. Experiments across three MLLM families and four datasets show that SAE++ improves hierarchical coherence and enables effective concept-level steering. Future work includes extending the framework to deeper hierarchies, additional modalities, and language-side representations. See Appendix A for detailed limitations such as reliance on Level-1 SAE quality.

Acknowledgments and Disclosure of Funding

We thank all reviewers, the area chair, and the senior area chair for their valuable comments.

Funding. HW is supported by the Amazon Faculty Research Award, Microsoft AI & Society Fellowship, Nokia Faculty Research Award, NSF CAREER Award IIS-2340125, NIH grant R01CA297832, and NSF grant IIS-2127918.

Competing interests. The authors declare no competing interests.

References

[1] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *CoRR*, abs/2308.12966, 2023.

[3] Gemma Team. Gemma: Open models based on gemini research and technology. *CoRR*, abs/2403.08295, 2024.

[4] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pages 8469–8488. PMLR, 2023.

[5] Michael Moor, Oishi Banerjee, Zahra Shakeri Hossein Abad, Harlan M. Krumholz, Jure Leskovec, Eric J. Topol, and Pranav Rajpurkar. Foundation models for generalist medical artificial intelligence. *Nature*, 616(7956):259–265, 2023.

[6] Zechen Bai, Pichao Wang, Tianjun Xiao, Tong He, Zongbo Han, Zheng Zhang, and Mike Zheng Shou. Hallucination of multimodal large language models: A survey. *CoRR*, abs/2404.18930, 2024.

[7] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 292–305, 2023.

[8] Xiangyu Qi, Kaixuan Huang, Ashwinee Panda, Peter Henderson, Mengdi Wang, and Prateek Mittal. Visual adversarial examples jailbreak aligned large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 21527–21536, 2024.

[9] Siyuan Ma, Weidi Luo, Yu Wang, and Xiaogeng Liu. Visual-roleplay: Universal jailbreak attack on multimodal large language models via role-playing image character. *arXiv preprint arXiv:2405.20773*, 2024.

[10] Jingquan Yan and Hao Wang. Self-interpretable time series prediction with counterfactual explanations. In *ICML*, 2023.

[11] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022.

[12] Yossi Gandelsman, Alexei Efros, and Jacob Steinhardt. Interpreting clip's image representation via text-based decomposition. In *International Conference on Learning Representations*, volume 2024, pages 18395–18416, 2024.

[13] Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick 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 Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

[14] Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[15] Hyesu Lim, Jinho Choi, Jaegul Choo, and Steffen Schneider. Sparse autoencoders reveal selective remapping of visual concepts during adaptation. In *International Conference on Learning Representations*, volume 2025, pages 24444–24469, 2025.

[16] Vladimir Zaigrajew, Hubert Baniecki, and Przemyslaw Biecek. Interpreting CLIP with hierarchical sparse autoencoders. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025.

[17] Hantao Lou, Changye Li, Jiaming Ji, and Yaodong Yang. SAE-V: interpreting multimodal models for enhanced alignment. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025.

[18] Shufan Shen, Junshu Sun, Qingming Huang, and Shuhui Wang. Vl-sae: Interpreting and enhancing vision-language alignment with a unified concept set. *Advances in Neural Information Processing Systems*, 38:45235–45265, 2026.

[19] Isabel Papadimitriou, Huangyuan Su, Thomas Fel, Naomi Saphra, Sham M. Kakade, and Stephanie Gil. Interpreting the linear structure of vision-language model embedding spaces. *CoRR*, abs/2504.11695, 2025.

[20] Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022.

[21] Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025.

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[23] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[24] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[25] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[26] Gemini Team. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805, 2023.

[27] Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261, 2025.

[28] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, James Molloy, Jilin Chen, Michael Isard, Paul Barham, Tom Hennigan, Ross McIlroy, Melvin Johnson, Johan Schalkwyk, Eli Collins, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Clemens Meyer, Gregory Thornton, Zhen Yang, Henryk Michalewski, Zaheer Abbas, Nathan Schucher, Ankesh Anand, Richard Ives, James Keeling, Karel Lenc, Salem Haykal, Siamak Shakeri, Pranav Shyam, Aakanksha Chowdhery, Roman Ring, Stephen Spencer, Eren Sezener, and et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *CoRR*, abs/2403.05530, 2024.

[29] Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024.

[30] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *CoRR*, abs/2409.12191, 2024.

[31] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *CoRR*, abs/2502.13923, 2025.

[32] Qwen Team. Qwen3-vl technical report. *CoRR*, abs/2511.21631, 2025.

[33] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, Yaofeng Sun, Chengqi Deng, Hanwei Xu, Zhenda Xie, and Chong Ruan. Deepseek-vl: Towards real-world vision-language understanding. *CoRR*, abs/2403.05525, 2024.

[34] Pravesh Agrawal, Szymon Antoniak, Emma Bou Hanna, Baptiste Bout, Devendra Singh Chaplot, Jessica Chudnovsky, Diogo Costa, Baudouin De Monicault, Saurabh Garg, Théophile Gervet, Soham Ghosh, Amélie Héliou, Paul Jacob, Albert Q. Jiang, Kartik Khandelwal, Timothée Lacroix, Guillaume Lample, Diego de Las Casas, Thibaut Lavril, Teven Le Scao, Andy Lo, William Marshall, Louis Martin, Arthur Mensch, Pavankumar Muddireddy, Valera Nemychnikova, Marie Pellat, Patrick von Platen, Nikhil Raghuraman, Baptiste Rozière, Alexandre Sablayrolles, Lucile Saulnier, Romain Sauvestre, Wendy Shang, Roman Soletskyi, Lawrence Stewart, Pierre Stock, Joachim Studnia, Sandeep Subramanian, Sagar Vaze, Thomas Wang, and Sophia Yang. Pixtral 12b. *CoRR*, abs/2410.07073, 2024.

[35] Gemma Team. Gemma 3 technical report. *CoRR*, abs/2503.19786, 2025.

[36] Meta AI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models, 2024.

[37] Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models. *Advances in Neural Information Processing Systems*, 37:83091–83118, 2024.

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

[39] Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving sparse decomposition of language model activations with gated sparse autoencoders. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[40] Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *CoRR*, abs/2407.14435, 2024.

[41] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.

[42] Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, 2024.

[43] Hengyi Wang, Shiwei Tan, and Hao Wang. Probabilistic conceptual explainers: Towards trustworthy conceptual explanations for vision foundation models. In *ICML*, 2024.

[44] Hengyi Wang, Shiwei Tan, Zhiqing Hong, Desheng Zhang, and Hao Wang. Variational language concepts for interpreting pretrained language models. *arXiv preprint*, 2024.

[45] Mateusz Pach, Shyamgopal Karthik, Quentin Bouniot, Serge Belongie, and Zeynep Akata. Sparse autoencoders learn monosemantic features in vision-language models. *Advances in Neural Information Processing Systems*, 38:95706–95742, 2026.

[46] Hengyi Wang, Shiwei Tan, and Hao Wang. Probabilistic conceptual explainers: Trustworthy conceptual explanations for vision foundation models. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pages 51502–51522. PMLR / OpenReview.net, 2024.

[47] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[48] Grant Van Horn, Elijah Cole, Sara Beery, Kimberly Wilber, Serge Belongie, and Oisin Mac Aodha. Benchmarking representation learning for natural world image collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12884–12893, 2021.

[49] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[50] Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *CoRR*, abs/2412.06410, 2024.

[51] David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. *CoRR*, abs/1609.09106, 2016.

A Limitations

Scope of evaluation. Our experiments focus on hierarchical visual concept discovery in MLLMs. We evaluate SAE++ on three model families and four visual datasets, but the current study does not fully cover other modalities, such as audio, video, or language-side reasoning representations. Moreover, our main quantitative evaluation uses HMS to measure whether Level-1 concepts under the same Level-2 parent are semantically coherent. HMS does not explicitly penalize duplicated concepts. Although HMS captures an important aspect of hierarchical interpretability, it does not exhaust all possible notions of a useful concept hierarchy. For example, different downstream tasks may require hierarchies organized by function, causality, compositionality, or task relevance rather than visual similarity alone. Extending SAE++ to additional modalities, tasks, and hierarchy metrics is an important direction for future work.

Dependence on Level-1 SAE quality and decoder-atom abstraction. SAE++ assumes that the Level-1 SAE learns meaningful low-level concept directions, and then trains the Level-2 SAE on the corresponding Level-1 decoder atoms. This design is effective when Level-1 decoder directions are interpretable and semantically organized, but the quality of the learned hierarchy can degrade if the Level-1 SAE contains noisy, dead, highly polysemantic, or poorly reconstructed features. The method also relies on the interpretation that decoder atoms can serve as inputs for higher-level abstraction. While our empirical results support this design, we do not exhaustively study how SAE++ performance changes with different Level-1 SAE architectures, sparsity levels, expansion ratios, target layers, or SAE training datasets. A more systematic analysis of these design choices would help establish robust defaults for applying SAE++ to new models.

Computation, steering, and evaluation limitations. SAE++ introduces additional training-time computation compared with a single flat SAE because it jointly trains a Level-2 SAE over Level-1 decoder atoms. This overhead is manageable in our experiments, but may become more significant for larger activation dimensions, larger SAE dictionaries, or deeper hierarchies with more than two levels. For concept steering, we evaluate a limited set of Level-2 concepts and use four LLM judges to estimate whether the target concept appears or is removed. This provides useful evidence that SAE++ concepts can support group-level interventions, but it is not a complete measure of causal control or safety. Future work should evaluate steering over more concepts, more prompts, human or task-specific judgments, and possible side effects of interventions on unrelated visual concepts.

B Theoretical Analysis

This appendix provides the complete assumptions and proofs for the theoretical claims in Sec. 4. We first prove the sparse-bottleneck failure of naive stacked SAEs, and then prove the shared-prefix error amplification result for Matryoshka SAEs. The bottleneck result concerns uniform sparse-code reconstruction, and the shared-prefix result uses an orthonormal linear surrogate. Note that these are structural analyses rather than guarantees for non-convex training.

B.1 Impossibility of Naive Stacked SAE Bottlenecks

We analyze a stacked SAE in which a first SAE produces a sparse code in $\mathbb{R}^n$, and a second SAE compresses this code through a smaller and sparser bottleneck in $\mathbb{R}^m$. Let

$$\mathcal{S}_{n,k_1}(B) := \{\mathbf{z} \in \mathbb{R}^n : \|\mathbf{z}\|_0 \le k_1,\ \|\mathbf{z}\|_2 \le B\}$$

denote the set of possible Level-1 sparse codes. A stacked SAE introduces an encoder-decoder pair

$$g : \mathbb{R}^n \to \mathbb{R}^m, \qquad h : \mathbb{R}^m \to \mathbb{R}^n,$$

with the goal of reconstructing $\mathbf{z} \in \mathcal{S}_{n,k_1}(B)$ from $g(\mathbf{z})$.

We assume:

1. **Uniform reconstruction:**

$$\sup_{\mathbf{z} \in \mathcal{S}_{n,k_1}(B)} \|h(g(\mathbf{z})) - \mathbf{z}\|_2 \le \varepsilon.$$

2. **Sparse bottleneck:**

$$g(\mathbf{z}) \in \mathcal{S}_{m,k_2}(R) \quad \text{for all } \mathbf{z} \in \mathcal{S}_{n,k_1}(B), \qquad k_2 < k_1.$$

3. **Decoder regularity:** h is L_h-Lipschitz:

$$\|h(\mathbf{u}) - h(\mathbf{v})\|_2 \le L_h \|\mathbf{u} - \mathbf{v}\|_2.$$

B.1.1 Auxiliary Lemmas

Definition B.1 (ρ-separated set)**.** *Let $(\mathcal{X}, \|\cdot\|_2)$ be a metric space. A set $\mathcal{P} \subset \mathcal{X}$ is ρ-separated if*

$$\|\mathbf{x} - \mathbf{y}\|_2 \ge \rho \quad \textit{for all distinct } \mathbf{x}, \mathbf{y} \in \mathcal{P}.$$

Lemma B.1 (Uniform reconstruction implies bottleneck separation)**.** *If*

$$\sup_{\mathbf{z} \in \mathcal{S}_{n,k_1}(B)} \|h(g(\mathbf{z})) - \mathbf{z}\|_2 \le \varepsilon,$$

then for all $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{S}_{n,k_1}(B)$,

$$\|g(\mathbf{z}_1) - g(\mathbf{z}_2)\|_2 \ge \frac{\|\mathbf{z}_1 - \mathbf{z}_2\|_2 - 2\varepsilon}{L_h}.$$

Proof. By the triangle inequality,

$$\begin{aligned} \|\mathbf{z}_1 - \mathbf{z}_2\|_2 \le \|\mathbf{z}_1 - h(g(\mathbf{z}_1))\|_2 + \|h(g(\mathbf{z}_1)) - h(g(\mathbf{z}_2))\|_2 \\ + \|h(g(\mathbf{z}_2)) - \mathbf{z}_2\|_2. \end{aligned} \tag{13}$$

The first and third terms are at most ε, and the middle term is at most $L_h\|g(\mathbf{z}_1) - g(\mathbf{z}_2)\|_2$. Rearranging proves the result. □

Lemma B.2 (A separated support packing for sparse codes)**.** *For $1 \le k_1 \le n$, $B > 0$, and $\rho \in (0, B]$, define*

$$r_\rho = \left\lceil \frac{k_1 \rho^2}{2B^2} \right\rceil - 1, \qquad V_{n,k_1}(r) = \sum_{i=0}^{r} \binom{k_1}{i} \binom{n - k_1}{i},$$

with binomial coefficients outside their valid range interpreted as zero. There is a ρ-separated set $\mathcal{P} \subset \mathcal{S}_{n,k_1}(B)$ satisfying

$$|\mathcal{P}| \ge \frac{\binom{n}{k_1}}{V_{n,k_1}(r_\rho)}.$$

Proof. For each k_1-element support $S \subset [n]$, let $\mathbf{z}_S = (B/\sqrt{k_1})\mathbf{1}_S$. Each vector has norm B and belongs to $\mathcal{S}_{n,k_1}(B)$. Greedily select a support S and remove every remaining support T with $|S \setminus T| \le r_\rho$. Each selection removes at most $V_{n,k_1}(r_\rho)$ supports: for a fixed i, there are $\binom{k_1}{i}\binom{n-k_1}{i}$ supports formed by replacing i indices. The selected family therefore has size at least the stated ratio. Any two selected supports satisfy $|S \setminus T| \ge r_\rho + 1$, and hence

$$\|\mathbf{z}_S - \mathbf{z}_T\|_2^2 = \frac{2B^2}{k_1}|S \setminus T| \ge \frac{2B^2}{k_1}\left\lceil \frac{k_1\rho^2}{2B^2} \right\rceil \ge \rho^2.$$

Thus the constructed vectors are separated even when their supports overlap. □

Lemma B.3 (Packing upper bound for sparse bottleneck codes)**.** *Let*

$$\mathcal{S}_{m,k_2}(R) = \{\mathbf{u} \in \mathbb{R}^m : \|\mathbf{u}\|_0 \le k_2,\ \|\mathbf{u}\|_2 \le R\}.$$

Any δ-separated subset $\mathcal{Q} \subset \mathcal{S}_{m,k_2}(R)$ satisfies

$$|\mathcal{Q}| \le \binom{m}{k_2}\left(1 + \frac{2R}{\delta}\right)^{k_2}.$$

Proof. Decompose $\mathcal{S}_{m,k_2}(R)$ by supports. For each $T \subset [m]$ with $|T| = k_2$, define

$$\mathcal{S}_T(R) := \{\mathbf{u} \in \mathbb{R}^m : \operatorname{supp}(\mathbf{u}) \subseteq T,\ \|\mathbf{u}\|_2 \le R\}.$$

Then

$$\mathcal{S}_{m,k_2}(R) \subseteq \bigcup_{T\subset[m],\,|T|=k_2} \mathcal{S}_T(R).$$

For a δ-separated set $\mathcal{Q}$, let $\mathcal{Q}_T = \mathcal{Q} \cap \mathcal{S}_T(R)$. Then

$$|\mathcal{Q}| \le \sum_{T:\,|T|=k_2} |\mathcal{Q}_T|.$$

For fixed T, $\mathcal{S}_T(R)$ is isometric to a k_2-dimensional Euclidean ball of radius R. If $Q' \subset B_{k_2}(R)$ is δ-separated, then balls of radius $\delta/2$ centered at points in Q' are disjoint and contained in $B_{k_2}(R+\delta/2)$. A volume comparison gives

$$|Q'|\,\mathrm{Vol}(B_{k_2}(\delta/2)) \le \mathrm{Vol}(B_{k_2}(R+\delta/2)),$$

and hence

$$|Q'| \le \left(\frac{R+\delta/2}{\delta/2}\right)^{k_2} = \left(1+\frac{2R}{\delta}\right)^{k_2}.$$

Summing over the $\binom{m}{k_2}$ supports proves the claim. □

B.1.2 Main Impossibility Result

Theorem B.1 (Failure of stacked SAEs under a support-packing capacity condition)**.** *Assume uniform reconstruction, a k_2-sparse bottleneck with $k_2 < k_1$, and an L_h-Lipschitz decoder, with $L_h > 0$ and $\varepsilon \le B/8$. Let $r_* = \lceil k_1/8 \rceil - 1$ and use the support-neighborhood count V_{n,k_1} from Lemma B.2. If*

$$\log\binom{m}{k_2} + k_2 \log\left(1+\frac{8RL_h}{B}\right) < \log\binom{n}{k_1} - \log V_{n,k_1}(r_*), \tag{14}$$

then no such encoder-decoder pair can uniformly reconstruct $\mathcal{S}_{n,k_1}(B)$.

Proof. Set $\rho = B/2$. Lemma B.2 gives a ρ-separated set $\mathcal{P}$ with

$$\log|\mathcal{P}| \ge \log\binom{n}{k_1} - \log V_{n,k_1}(r_*). \tag{15}$$

By Lemma B.1, $g(\mathcal{P})$ is δ-separated, where

$$\delta = \frac{\rho - 2\varepsilon}{L_h} \ge \frac{B}{4L_h} > 0.$$

The bottleneck packing upper bound gives

$$\log|g(\mathcal{P})| \le \log\binom{m}{k_2} + k_2 \log\left(1+\frac{8RL_h}{B}\right). \tag{16}$$

Separation makes g injective on $\mathcal{P}$, so $|g(\mathcal{P})| = |\mathcal{P}|$. Combining these two bounds contradicts Eq. (14). □

Remark B.1 (Relation to the empirical sparsity configuration)**.** *For fixed k_1 and sufficiently large n, the support-packing term grows as $(k_1 - r_*)\log n + O(1)$. With $m < n$ and bounded RL_h/B, the bottleneck term is at most $k_2 \log n + O(1)$. Thus the sufficient condition holds for large n if $k_2 < k_1 - r_*$. In particular, for the empirical $k_1 = 6$, $k_2 = 1$, we have $r_* = 0$ and $V_{n,6}(0) = 1$: distinct equal-amplitude six-element supports already form a $B/2$-separated packing. The result concerns uniform reconstruction under the stated assumptions.*

B.2 Full Proofs for Shared-Prefix Error Amplification

We now prove the shared-prefix amplification result used in Sec. 4.2. The analysis is idealized: it uses an orthonormal linear surrogate to isolate the structural effect of nested-prefix reuse, rather than characterizing nonlinear SAE training dynamics.

B.2.1 Preliminaries

For any linear subspace $U \subseteq \mathbb{R}^d$, let P_U denote the orthogonal projector onto U, and let $\|\cdot\|_F$ denote the Frobenius norm.

Lemma B.4 (Projector-distance identity)**.** *Let $A, B \subseteq \mathbb{R}^d$ be finite-dimensional subspaces. Then*

$$\|P_A - P_B\|_F^2 = \dim(A) + \dim(B) - 2\,\mathrm{Tr}(P_A P_B).$$

In particular, if $\dim(A) = \dim(B) = m$, then

$$\|P_A - P_B\|_F^2 = 2m - 2\,\mathrm{Tr}(P_A P_B).$$

Proof. Since orthogonal projectors are symmetric and idempotent,

$$\begin{aligned}
\|P_A - P_B\|_F^2 &= \mathrm{Tr}\big((P_A - P_B)^2\big) && (17)\\
&= \mathrm{Tr}(P_A^2) + \mathrm{Tr}(P_B^2) - 2\,\mathrm{Tr}(P_A P_B) && (18)\\
&= \mathrm{Tr}(P_A) + \mathrm{Tr}(P_B) - 2\,\mathrm{Tr}(P_A P_B) && (19)\\
&= \dim(A) + \dim(B) - 2\,\mathrm{Tr}(P_A P_B). && (20)
\end{aligned}$$

The equal-dimensional case follows immediately. □

Lemma B.5 (One-dimensional projector distance)**.** *Let A and B be one-dimensional subspaces of $\mathbb{R}^d$. Then*

$$\|P_A - P_B\|_F^2 \le 2.$$

Proof. Let $A = \mathrm{span}\{a\}$ and $B = \mathrm{span}\{b\}$, where a, b are unit vectors. Then $P_A = aa^\top$, $P_B = bb^\top$, and

$$\mathrm{Tr}(P_A P_B) = (a^\top b)^2.$$

By Lemma B.4,

$$\|P_A - P_B\|_F^2 = 2 - 2(a^\top b)^2 \le 2.$$

□

Lemma B.6 (Shared-prefix reuse count)**.** *For nested prefix sizes $1 \le m_1 < \cdots < m_L$, the decoder direction at position p is contained in exactly*

$$C_{\mathrm{MSAE}}(p) := |\{t : m_t \ge p\}|$$

prefixes. If $m_{\ell-1} < p \le m_\ell$, with $m_0 := 0$, then

$$C_{\mathrm{MSAE}}(p) = L - \ell + 1.$$

In particular, $p \le m_{L-1}$ implies $C_{\mathrm{MSAE}}(p) \ge 2$.

Proof. The t-th prefix contains exactly the first m_t decoder directions. Therefore, the direction at position p is included in prefix t if and only if $p \le m_t$. Hence the number of prefixes containing that direction is

$$C_{\mathrm{MSAE}}(p) = |\{t : m_t \ge p\}|.$$

If $m_{\ell-1} < p \le m_\ell$, the direction first appears in prefix ℓ and is reused by prefixes $\ell, \ell+1, \ldots, L$, so

$$C_{\mathrm{MSAE}}(p) = L - \ell + 1.$$

If $p \le m_{L-1}$, then $\ell \le L - 1$, hence $C_{\mathrm{MSAE}}(p) \ge 2$. □

B.2.2 From Matryoshka Prefix Reconstruction to Semantic Mismatch

Recall that Matryoshka SAEs implement hierarchy by training nested index subsets

$$\mathcal{I}_t = \{1, \dots, m_t\}, \qquad m_1 < \cdots < m_L,$$

under the prefix reconstruction objective

$$\sum_{t=1}^{L} \left\| \mathbf{x} - \left(\mathbf{W}_{\text{dec}}^{(:,\mathcal{I}_t)} \mathbf{f}_{\mathcal{I}_t} + \mathbf{b} \right) \right\|_2^2.$$

Thus, the t-th prefix is encouraged to reconstruct the input using only the first m_t decoder columns.

To isolate the effect of prefix reuse, we consider an orthonormal linear surrogate of the Matryoshka decoder. Let

$$D = [\mathbf{d}_1, \dots, \mathbf{d}_{m_L}] \in \mathbb{R}^{d \times m_L}$$

be the ordered decoder directions of the largest prefix, where $\mathbf{d}_p$ corresponds to the p-th decoder column. The t-th prefix spans the subspace

$$U_t(D) = \text{span}\{\mathbf{d}_1, \dots, \mathbf{d}_{m_t}\}, \qquad 1 \le m_1 < \cdots < m_L.$$

Under the orthonormal linear surrogate, the nonlinear decoder restricted to prefix t is replaced by the orthogonal projection onto $U_t(D)$. Thus, prefix t reconstructs $\mathbf{x}$ as $P_{U_t(D)}\mathbf{x}$, and the Matryoshka prefix reconstruction objective is idealized as

$$\sum_{t=1}^{L} \mathbb{E}_{\mathbf{x}} \left[\|\mathbf{x} - P_{U_t(D)}\mathbf{x}\|_2^2 \right].$$

This is the linear surrogate corresponding to the prefix reconstruction loss in the main text; it is used only to isolate shared-prefix reuse, not to model the full nonlinear SAE training dynamics.

We then evaluate semantic alignment by comparing the prefix subspace $U_t(D)$ with an ideal semantic target. Let $\mathcal{M}_t \subseteq \mathbb{R}^d$ denote the target semantic subspace for level t, i.e., the span of the semantic directions that the t-th prefix is intended to capture. We set $\dim(\mathcal{M}_t) = m_t$, matching the dimension of $U_t(D)$. The prefix semantic mismatch is

$$\mathcal{E}_t^{\text{MSAE}}(D) := \|P_{U_t(D)} - P_{\mathcal{M}_t}\|_F^2,$$

and the total prefix semantic mismatch is

$$\mathcal{E}_{\text{MSAE}}^{\text{tot}}(D) := \sum_{t=1}^{L} \mathcal{E}_t^{\text{MSAE}}(D).$$

This projector mismatch is not the original reconstruction objective. It is a semantic diagnostic defined on top of the linear surrogate: it measures whether the subspace used by prefix t aligns with the semantic subspace that level t is intended to represent.

B.2.3 Normalized Local Semantic Error

We now define the local semantic errors used in the theorem. Suppose there are q nuisance decoder directions in the Matryoshka ordering. Formally, write

$$\mathbf{d}_{p_i} = \mathbf{n}_i, \qquad i = 1, \dots, q,$$

where the $\mathbf{n}_i$'s are mutually orthonormal and each $\mathbf{n}_i$ is orthogonal to every target semantic subspace $\mathcal{M}_t$. For every affected prefix t with $m_t \ge p_i$, define

$$U_t^{(-i)}(D) := U_t(D) \cap \text{span}\{\mathbf{n}_i\}^{\perp}.$$

This removes the one-dimensional nuisance direction $\mathbf{n}_i$ from the prefix subspace.

Semantic repair direction assumption. For every nuisance direction $\mathbf{n}_i$ and affected prefix t with $m_t \geq p_i$, assume there exists a unit vector

$$\mathbf{s}_{i,t} \in \mathcal{M}_t \cap \big(U_t^{(-i)}(D)\big)^{\perp}.$$

Define the repaired prefix subspace

$$\widetilde{U}_t^{(-i)}(D) := U_t^{(-i)}(D) \oplus \operatorname{span}\{\mathbf{s}_{i,t}\}.$$

This repair is an analysis device: after removing a non-semantic direction, it fills the freed one-dimensional slot with a valid semantic direction while preserving the prefix dimension.

The local Matryoshka semantic error caused by $\mathbf{n}_i$ in prefix t is

$$e_{i,t}^{\mathrm{MSAE}} := \mathcal{E}_t^{\mathrm{MSAE}}(D) - \|P_{\widetilde{U}_t^{(-i)}(D)} - P_{\mathcal{M}_t}\|_F^2.$$

Since the same decoder direction may appear in several prefixes, we normalize by its reuse count:

$$\overline{\mathcal{E}}_{\mathrm{MSAE}} := \sum_{i=1}^{q} \frac{1}{C_{\mathrm{MSAE}}(p_i)} \sum_{t:m_t \geq p_i} e_{i,t}^{\mathrm{MSAE}}.$$

For SAE++, each Level-1 atom $\mathbf{w}_j$ has an ideal semantic parent, denoted by $c(j)$. Let $\mathcal{P}_{c(j)}$ be the one-dimensional subspace corresponding to this ideal parent, and let $\widehat{\mathcal{P}}_j$ be the parent subspace assigned by the Level-2 code of $\mathbf{w}_j$. For a locally erroneous atom assignment j, define

$$e_j^{\mathrm{SAE++}} := \|P_{\widehat{\mathcal{P}}_j} - P_{\mathcal{P}_{c(j)}}\|_F^2.$$

For a set $\mathcal{B}$ of locally erroneous atom assignments, define

$$\overline{\mathcal{E}}_{\mathrm{SAE++}}(\mathcal{B}) := \sum_{j \in \mathcal{B}} e_j^{\mathrm{SAE++}}.$$

No prefix-reuse normalization is needed for SAE++, because each atom-level error appears in only one atom-wise Level-2 term.

Theorem B.2 (Prefix reuse amplifies Matryoshka semantic error)**.** *Under the setup above, the total prefix semantic mismatch satisfies*

$$\mathcal{E}_{\mathrm{MSAE}}^{\mathrm{tot}}(D) \geq 2 \sum_{i=1}^{q} C_{\mathrm{MSAE}}(p_i).$$

Moreover, under the normalized local comparison,

$$\overline{\mathcal{E}}_{\mathrm{MSAE}} \geq 2q.$$

For SAE++, if $|\mathcal{B}| \leq q$, then

$$\overline{\mathcal{E}}_{\mathrm{SAE++}}(\mathcal{B}) \leq 2q \leq \overline{\mathcal{E}}_{\mathrm{MSAE}}.$$

Proof. We first prove the total prefix semantic mismatch bound. Fix a prefix t, and let

$$I_t := \{i : m_t \geq p_i\}, \qquad q_t := |I_t|.$$

The set I_t indexes the nuisance directions contained in $U_t(D)$. Because these nuisance directions are mutually orthonormal and orthogonal to $\mathcal{M}_t$, they occupy q_t orthogonal dimensions of $U_t(D)$ with zero overlap with $\mathcal{M}_t$. Therefore the trace overlap between $U_t(D)$ and $\mathcal{M}_t$ is at most $m_t - q_t$:

$$\operatorname{Tr}(P_{U_t(D)} P_{\mathcal{M}_t}) \leq m_t - q_t.$$

Using Lemma B.4,

$$\|P_{U_t(D)} - P_{\mathcal{M}_t}\|_F^2 = 2m_t - 2\operatorname{Tr}(P_{U_t(D)} P_{\mathcal{M}_t}) \geq 2q_t.$$

Summing over all prefixes gives

$$\mathcal{E}_{\mathrm{MSAE}}^{\mathrm{tot}}(D) = \sum_{t=1}^{L} \|P_{U_t(D)} - P_{\mathcal{M}_t}\|_F^2 \geq \sum_{t=1}^{L} 2q_t.$$

Since

$$\sum_{t=1}^{L} q_t = \sum_{t=1}^{L} |\{i : m_t \geq p_i\}| = \sum_{i=1}^{q} |\{t : m_t \geq p_i\}| = \sum_{i=1}^{q} C_{\mathrm{MSAE}}(p_i),$$

we obtain

$$\mathcal{E}^{\mathrm{tot}}_{\mathrm{MSAE}}(D) \geq 2 \sum_{i=1}^{q} C_{\mathrm{MSAE}}(p_i).$$

Next, we prove the normalized Matryoshka local-error bound. Fix a nuisance direction $\mathbf{n}_i$ and an affected prefix t with $m_t \geq p_i$. Let

$$U := U_t(D), \qquad \widetilde{U} := \widetilde{U}_t^{(-i)}(D), \qquad M := \mathcal{M}_t.$$

Both U and $\widetilde{U}$ have dimension m_t, and M also has dimension m_t. By Lemma B.4,

$$e^{\mathrm{MSAE}}_{i,t} = 2\,\mathrm{Tr}(P_{\widetilde{U}} P_M) - 2\,\mathrm{Tr}(P_U P_M).$$

The subspaces U and $\widetilde{U}$ differ only by replacing $\mathbf{n}_i$ with $\mathbf{s}_{i,t}$. The shared component $U_t^{(-i)}(D)$ contributes equally to both trace terms and cancels. Because $\mathbf{n}_i \perp M$, the removed direction contributes zero trace overlap with M. Because $\mathbf{s}_{i,t} \in M$ is a unit vector, the repair direction contributes one unit of trace overlap with M. Therefore

$$\mathrm{Tr}(P_{\widetilde{U}} P_M) - \mathrm{Tr}(P_U P_M) = 1,$$

and hence

$$e^{\mathrm{MSAE}}_{i,t} = 2.$$

Averaging over the $C_{\mathrm{MSAE}}(p_i)$ affected prefixes gives

$$\frac{1}{C_{\mathrm{MSAE}}(p_i)} \sum_{t : m_t \geq p_i} e^{\mathrm{MSAE}}_{i,t} = 2.$$

Summing over $i = 1, \ldots, q$ yields

$$\overline{\mathcal{E}}_{\mathrm{MSAE}} = 2q,$$

and in particular $\overline{\mathcal{E}}_{\mathrm{MSAE}} \geq 2q$.

Finally, for SAE++, each $e_j^{\mathrm{SAE++}}$ is the Frobenius distance between two one-dimensional projectors. By Lemma B.5,

$$e_j^{\mathrm{SAE++}} \leq 2.$$

Thus, if $|\mathcal{B}| \leq q$,

$$\overline{\mathcal{E}}_{\mathrm{SAE++}}(\mathcal{B}) = \sum_{j \in \mathcal{B}} e_j^{\mathrm{SAE++}} \leq 2|\mathcal{B}| \leq 2q.$$

Combining the inequalities gives

$$\overline{\mathcal{E}}_{\mathrm{SAE++}}(\mathcal{B}) \leq 2q \leq \overline{\mathcal{E}}_{\mathrm{MSAE}}.$$

□

Remark B.2 (Scope of the idealization)**.** *The result above is not a universal claim about nonlinear Matryoshka SAE optimization. It isolates one structural effect of nested prefixes under an orthonormal linear surrogate: a semantic error in an early decoder direction is reused by every downstream prefix containing that direction. The normalized comparison removes this repeated counting, while the total prefix semantic mismatch keeps it.*

B.3 Cluster-consistent Coding Compatibility

The main text focuses on avoiding stacked bottlenecks and shared-prefix amplification. For completeness, we also record a simple representational compatibility property of the SAE++ second stage.

Proposition B.1 (Cluster-consistent coding compatibility)**.** *Assume $K \le n_2$ and that each input atom has a unique semantic parent. Under the clustered semantic atom model*

$$\mathbf{w}_j = \boldsymbol{\mu}_{c(j)} + \boldsymbol{\xi}_j, \qquad \|\boldsymbol{\xi}_j\|_2 \le \sigma.$$

Then there exists a Level-2 sparse encoder g_2 such that atoms with the same semantic parent have identical Level-2 support. Consequently, for every nonempty subset $S \subseteq \{1, \dots, n_1\}$ with at least one same-parent pair,

$$\mathcal{E}^{\text{supp}}_{\text{SAE++}}(S) = 0.$$

Proof. For each semantic prototype index $c \in \{1, \dots, K\}$, choose one Level-2 latent unit and denote its one-hot code by $\mathbf{e}_c$. Define

$$g_2(\mathbf{w}_j) := \mathbf{e}_{c(j)}.$$

If $c(i) = c(j)$, then

$$g_2(\mathbf{w}_i) = \mathbf{e}_{c(i)} = \mathbf{e}_{c(j)} = g_2(\mathbf{w}_j),$$

so

$$\text{supp}(g_2(\mathbf{w}_i)) = \text{supp}(g_2(\mathbf{w}_j)).$$

Thus every same-parent pair contributes zero to the support-inconsistency indicator, and hence

$$\mathcal{E}^{\text{supp}}_{\text{SAE++}}(S) = 0.$$

This is an achievability statement about representational compatibility, not an optimization guarantee. □

C Generalization to an L-Level SAE++

C.1 General L-Level SAE++

While the description in Sec. 3.2 focuses on a two-level hierarchy for clarity, the proposed SAE++ framework naturally extends to an arbitrary number of hierarchical levels. In this section, we describe the general formulation of an L-level cascaded sparse autoencoder, which enables progressively higher-order semantic abstraction.

Architecture Overview. Let Level-0 denote the original LLM activation space, with input activations $\mathbf{x} \in \mathbb{R}^d$. The model consists of L SAEs arranged in a cascade. Level 1 operates on the original activations. Each Level-ℓ SAE ($\ell = 2, \dots, L$) operates on the decoder weights learned at Level-$(\ell - 1)$, thereby inducing a hierarchy of concepts from low-level concepts to high-level semantic structures.

We denote by n_ℓ the number of latent units at Level ℓ, and by

$$\mathbf{W}^{(\ell)}_{\text{dec}} = [\mathbf{w}^{(\ell)}_1, \dots, \mathbf{w}^{(\ell)}_{n_\ell}] \tag{21}$$

the decoder matrix of the Level-ℓ SAE, where each column $\mathbf{w}^{(\ell)}_j \in \mathbb{R}^d$ represents a concept direction in the original activation space.

Level-ℓ SAE. For $\ell \ge 2$, the Level-ℓ SAE consists of an encoder-decoder pair (g_ℓ, h_ℓ). Its input is given by the decoder weight columns of the previous level, $\{\mathbf{w}^{(\ell-1)}_j\}_{j=1}^{n_{\ell-1}}$. For each such weight column, the forward pass is defined as

$$\mathbf{f}^{(\ell)} = g_\ell(\mathbf{w}^{(\ell-1)}_j) = \sigma\Big(\mathbf{W}^{(\ell)}_{\text{enc}}(\mathbf{w}^{(\ell-1)}_j - \mathbf{b}^{(\ell)})\Big), \tag{22}$$

$$\widehat{\mathbf{w}}^{(\ell-1)}_j = h_\ell(\mathbf{f}^{(\ell)}) = \mathbf{W}^{(\ell)}_{\text{dec}}\mathbf{f}^{(\ell)} + \mathbf{b}^{(\ell)}. \tag{23}$$

Here, $\mathbf{W}^{(\ell)}_{\text{dec}}$ encodes the concept directions at Level ℓ, while the sparse code $\mathbf{f}^{(\ell)} \in \mathbb{R}^{n_\ell}$ represents the assignment of a lower-level concept to higher-level abstractions.

Level-ℓ Objective. The training objective for the Level-ℓ SAE ($\ell \geq 2$) is defined as

$$\mathcal{L}_\ell\Big(\mathbf{W}_{\text{dec}}^{(\ell-1)}, \boldsymbol{\theta}_\ell\Big) = \frac{1}{n_{\ell-1}} \sum_{j=1}^{n_{\ell-1}} \Big(\big\|\mathbf{w}_j^{(\ell-1)} - h_\ell(g_\ell(\mathbf{w}_j^{(\ell-1)}))\big\|_2^2 + \lambda_\ell \mathcal{S}(g_\ell(\mathbf{w}_j^{(\ell-1)}))\Big), \tag{24}$$

where $\mathcal{S}(\cdot)$ is typically an ℓ_1 sparsity penalty, and $\boldsymbol{\theta}_\ell = \{\mathbf{W}_{\text{enc}}^{(\ell)}, \mathbf{W}_{\text{dec}}^{(\ell)}, \mathbf{b}^{(\ell)}\}$.

Joint End-to-End Optimization. The full L-level SAE++ is trained end-to-end with a weighted sum of reconstruction and sparsity objectives across all levels:

$$\mathcal{L}_{\text{final}} = \mathbb{E}_{\mathbf{x}}[\mathcal{L}_1(\mathbf{x}, \theta_1)] + \sum_{\ell=2}^{L} \alpha_\ell \, \mathcal{L}_\ell\Big(\mathbf{W}_{\text{dec}}^{(\ell-1)}, \theta_\ell\Big), \tag{25}$$

where α_ℓ controls the relative importance of semantic abstraction at Level ℓ.

Dynamic Masking of Dead Latents (General L-Level Case). To mitigate computational inefficiency arising from inactive ("dead") latents in deep hierarchies, we generalize the dynamic masking strategy to all higher levels of SAE++. At each training step t with mini-batch $\mathcal{B}$, the Level-ℓ SAE ($\ell \geq 2$) is optimized exclusively over the subset of Level-$(\ell-1)$ decoder atoms that are active under the current forward pass of the immediately preceding level, while dead latents are ignored.

For Level $\ell = 2$, the active set is defined using the original LLM activations $\mathbf{x}$:

$$\mathcal{M}_t^{(2)} = \left\{ \mathbf{w}_j^{(1)} \;\middle|\; \sum_{\mathbf{x} \in \mathcal{B}} [g_1(\mathbf{x})]_j > 0 \right\}. \tag{26}$$

For higher levels $\ell > 2$, the activity of Level-$(\ell-1)$ latents is instead determined by the encoder of the Level-$(\ell-1)$ SAE applied to Level-$(\ell-2)$ decoder weight columns $\mathbf{W}_{\text{dec}}^{(\ell-2)} = [\mathbf{w}_1^{(\ell-2)}, \ldots, \mathbf{w}_{n_{\ell-2}}^{(\ell-2)}]$:

$$\mathcal{M}_t^{(\ell)} = \left\{ \mathbf{w}_j^{(\ell-1)} \;\middle|\; \sum_{\mathbf{w}^{(\ell-2)} \in \mathbf{W}_{\text{dec}}^{(\ell-2)}} \big[g_{\ell-1}(\mathbf{w}^{(\ell-2)})\big]_j > 0 \right\}. \tag{27}$$

Here, $[g_{\ell-1}(\cdot)]_j$ denotes the j-th entry of the Level-$(\ell-1)$ encoder output. By restricting each Level-ℓ objective to $\mathcal{M}_t^{(\ell)}$, we ensure that higher-level concepts are learned strictly from valid, active semantic directions at the preceding level, improving both computational efficiency and hierarchical semantic grounding in deep cascades.

Hierarchical Interpretation. Under this formulation, each Level-ℓ SAE induces a parent assignment mapping from Level-$(\ell-1)$ concepts to Level-ℓ concepts via the encoder activations $g_\ell(\cdot)$. Stacking multiple levels therefore yields a deep hierarchy of increasingly abstract, sparse, and semantically organized concepts, with Level-1 capturing atomic features and higher levels encoding progressively coarser semantic groupings.

This general formulation subsumes the two-level SAE++ as a special case with $L = 2$, while providing a principled and scalable framework for learning deep hierarchical concept structures from multimodal LLM activations.

C.2 Hierarchical Mono-Semanticity (HMS) Score: General L-Level Formulation

To evaluate the semantic coherence of hierarchical concepts discovered by SAE++ at multiple abstraction depths, we generalize the Hierarchical Mono-Semanticity (HMS) score from the two-level setting to an L-level hierarchy. This extension enables quantitative assessment of semantic alignment between concepts at adjacent levels throughout the hierarchy.

Level-1 Concept Representations. As in the two-level case, we begin by constructing semantic representations for Level-1 (lowest-level) SAE neurons. For each Level-1 hidden neuron j (i.e., each entry of $\mathbf{f}^{(1)}$), we define a semantic representation $\mathbf{R}_j^{(1)} \in \mathbb{R}^{d_e}$ using a set of N input images.

Let $\{\mathbf{E}_i\}_{i=1}^N$ denote image embeddings extracted from a fixed vision backbone, where $\mathbf{E}_i \in \mathbb{R}^{d_e}$ corresponds to image i. Let $v_{i,j}$ be the activation of Level-1 neuron j for the MLLM activation

associated with image i. We normalize activations using min–max normalization:

$$a_{i,j} = \frac{v_{i,j} - \min(\mathbf{v}_j)}{\max(\mathbf{v}_j) - \min(\mathbf{v}_j) + \epsilon}, \tag{28}$$

where $\mathbf{v}_j = \{v_{i,j}\}_{i=1}^{N}$. The Level-1 concept representation is then defined as

$$\mathbf{R}_j^{(1)} = \frac{\sum_{i=1}^{N} a_{i,j}\mathbf{E}_i}{\sum_{i=1}^{N} a_{i,j} + \epsilon}. \tag{29}$$

General L-Level HMS Definition. Assume an L-level hierarchical structure with Level-ℓ neurons indexed by $j_\ell \in \{1, \ldots, n_\ell\}$ for $\ell = 1, \ldots, L$. For each level $\ell \geq 2$, let $\pi_\ell(j_{\ell-1}) \in \{1, \ldots, n_\ell\}$ denote the parent assignment function that maps a Level-$(\ell-1)$ neuron to a Level-ℓ neuron. Each neuron at every level corresponds to a learned semantic concept.

For a given Level-ℓ neuron k, define its set of children at Level-$(\ell-1)$ as

$$\mathcal{C}_k^{(\ell)} = \{j \mid \pi_\ell(j) = k\}, \tag{30}$$

and let $m = |\mathcal{C}_k^{(\ell)}|$. Here $\epsilon > 0$ is a numerical stability constant.

The HMS score for a Level-ℓ neuron k is defined as the mean pairwise cosine similarity among the semantic representations of its Level-$(\ell-1)$ children:

$$\text{HMS}^{(\ell)}(k) = \frac{2}{m(m-1)} \sum_{\substack{u,v \in \mathcal{C}_k^{(\ell)} \\ u<v}} \frac{\left(\mathbf{R}_u^{(\ell-1)}\right)^\top \mathbf{R}_v^{(\ell-1)}}{\|\mathbf{R}_u^{(\ell-1)}\|_2 \, \|\mathbf{R}_v^{(\ell-1)}\|_2}. \tag{31}$$

An HMS score close to 1.0 indicates that the Level-ℓ neuron groups semantically coherent and monosemantic Level-$(\ell-1)$ concepts.

Semantic Representations at Higher Levels. For $\ell \geq 2$, the semantic representation of a Level-ℓ neuron k is defined recursively as the mean of its children's representations:

$$\mathbf{R}_k^{(\ell)} = \frac{1}{|\mathcal{C}_k^{(\ell)}|} \sum_{j \in \mathcal{C}_k^{(\ell)}} \mathbf{R}_j^{(\ell-1)}, \tag{32}$$

enabling HMS evaluation to propagate consistently across all levels of the hierarchy.

Instantiation for SAE++. In SAE++, the parent assignment functions $\{\pi_\ell\}_{\ell=2}^{L}$ are induced intrinsically by the cascaded SAEs. Specifically, for $\ell \geq 2$, each Level-$(\ell-1)$ neuron j is associated with its decoder weight column $\mathbf{w}_j^{(\ell-1)}$, which is passed through the Level-ℓ encoder g_ℓ. The parent index is defined as

$$\pi_\ell(j) = \operatorname*{argmax}_{r \in \{1, \ldots, n_\ell\}} \left[g_\ell(\mathbf{w}_j^{(\ell-1)})\right]_r. \tag{33}$$

Under this definition, the Level-ℓ HMS scores, $\text{HMS}^{(\ell)}$, directly quantify the semantic coherence of the hierarchical clustering learned end-to-end at each level of abstraction.

Instantiation for Baseline Architectures. For baseline models that do not explicitly define multi-level hierarchies, parent assignments $\pi_\ell(\cdot)$ are constructed post hoc using co-activation statistics between adjacent levels. The HMS score is then computed using the same general formulation above, enabling a unified and fair comparison across different SAE variants.

C.3 Key Differences between SAE++ and Hyper-Networks

Although SAE++ introduces a hierarchical structure over learned features, it is fundamentally different from hyper-network-based approaches [51]. Hyper-networks explicitly *generate or modulate* the parameters of a target network through a separate conditioning network, tightly coupling representation learning with parameter synthesis within a single forward pass.

In contrast, SAE++ treats the learned decoder atoms (i.e., weight columns) of the lower-level SAE as first-class *data* and applies sparse dictionary learning recursively to model semantic structure *among features themselves*. Importantly, the higher-level SAE does not act as a parameter generator or controller. Even though gradients are allowed to flow through $\mathcal{L}_2$ to the lower-level decoder in our final formulation, this interaction arises solely from a reconstruction-based objective over learned semantic directions, rather than from conditional parameterization.

As a result, SAE++ learns stable, reusable abstractions over low-level concept directions via sparsity and reconstruction constraints, yielding an explicit and interpretable concept hierarchy with well-defined parent-child relationships – distinct from the implicit, task-driven parameter conditioning characteristic of hyper-networks – and enabling principled evaluation through hierarchical mono-semanticity metrics.

D Additional Evaluation and Training Diagnostics

D.1 A Shared Parent-Assignment Control

To compare all methods under a common parent-assignment protocol, we replace their native assignments with average-linkage Hierarchical Agglomerative Clustering (HAC), using cosine distance on unit-normalized alive Level-1 decoder atoms. The number of clusters is fixed and shared across methods. We evaluate the resulting groups with the same HMS definition and semantic representations, excluding groups with fewer than two children.

Table 5: HMS under a shared HAC parent-assignment protocol. All native parent assignments are discarded. Higher is better; the best result in each row is bold.

MLLM	Data	BTK	TopK	ReLU	P-Ann.	GSAE	JReLU	MSAE	Stack.	SAE++
Qwen3	ImageNet	0.165	0.163	0.442	0.177	0.218	0.572	0.175	0.310	**0.703**
	COCO	0.119	0.124	0.049	0.223	0.117	0.347	0.706	0.157	**0.933**
Gemma-3	ImageNet	0.146	0.348	0.079	0.410	0.469	0.608	0.201	0.173	**0.722**
	COCO	0.215	0.266	0.058	0.181	0.260	0.306	0.787	0.192	**0.852**

As shown in Table 5, SAE++ achieves the highest HMS in all four settings under the shared HAC protocol. This control applies HAC to every trained dictionary, including SAE++, whereas the clustering ablation replaces its Level-2 SAE.

D.2 Cluster Sizes and Coverage

HMS is defined only for parents with at least two children. For the native evaluation, we report the number of these HMS-eligible parents and their child-count distribution. Atom coverage is the fraction of the complete Level-1 dictionary assigned to eligible parents, with dead atoms and atoms in singleton groups excluded from the numerator. The following statistics use ImageNet with Qwen3-VL-4B-Instruct and the native parent assignments, rather than the shared HAC control above.

Table 6: Native HMS-eligible cluster statistics on ImageNet with Qwen3-VL-4B-Instruct. Cluster-size statistics are computed over eligible clusters. Coverage is a proportion of the complete Level-1 dictionary.

Method	Eligible clusters	Size min/median/max	Atom coverage
BTK	143	2/68/4112	0.656
TopK	77	2/2/810	0.598
ReLU	18	3/8/922	0.219
P-Ann.	15	2/74/7774	0.547
GSAE	25	2/4/1048	0.522
JReLU	74	2/2/2083	0.230
MSAE	176	2/3/148	0.623
Stacked	249	2/8/14	0.101
SAE++	162	2/4/763	0.565

For SAE++, the 162 eligible clusters cover approximately 11,300 atoms, or 56.5% of the 20,000-atom dictionary. Their median and maximum sizes are four and 763. Since aggregate HMS weights each eligible parent equally, these statistics provide context for the reported coherence scores.

D.3 Steering Evaluation with Multiple Judges

We re-evaluate the same steering outputs with three additional LLM judges: Gemini-3-Flash, Nemotron-30B, and Opus-4.8, using the target-concept presence protocol of the original Gemini-2.5-Flash evaluation. NS denotes no steering, Rand. denotes the random steering baseline, and Single denotes steering the strongest individual Level-1 concept. Judge Avg. is the unweighted mean across the four judges, rounded to one decimal place.

Table 3 in the main text reports the insertion and suppression results for all four judges and their average.

D.3.1 Concept-level Variability and Paired Comparisons

Using Opus-4.8 judgments, we report the mean and standard deviation of success rates across the ten target concepts.

Table 7: Steering success rates (%): mean $\pm$ standard deviation across ten target concepts, judged by Opus-4.8.

Method	Insertion	Suppression
NS	4.7 ± 8.9	0.0 ± 0.0
Rand.	5.2 ± 10.0	9.6 ± 6.4
MSAE	41.0 ± 9.1	22.6 ± 7.1
Single	60.9 ± 12.2	30.3 ± 7.2
SAE++	72.6 ± 12.9	35.7 ± 7.8

We perform paired t-tests on the ten concept-level scores, pairing methods by target concept. Table 8 reports the resulting p-values; each comparison favors SAE++ with $p < 0.05$.

Table 8: Paired t-test p-values over ten matched concept-level scores (Opus-4.8).

SAE++ versus	Insertion	Suppression
NS	< 0.001	< 0.001
Rand.	< 0.001	< 0.001
Single	0.007	0.011
MSAE	< 0.001	0.001

D.3.2 Observed Off-target Interference

We sample 20 unrelated concepts and use Claude-Sonnet-5 with ten representative keywords per concept to judge their presence in captions. The off-target flip rate measures changes in concept-presence judgments between matched steered and unsteered captions. We report the mean and standard deviation across the 20 concepts as proportions.

Table 9: Off-target flip rates: mean $\pm$ standard deviation across 20 sampled unrelated concepts. Lower is better; NS is zero by construction.

Method	Insertion	Suppression
NS	0.000 ± 0.000	0.000 ± 0.000
Rand.	0.002 ± 0.002	0.001 ± 0.002
MSAE	0.005 ± 0.003	0.003 ± 0.002
Single	0.005 ± 0.002	0.003 ± 0.003
SAE++	0.004 ± 0.003	0.003 ± 0.004

SAE++ has off-target flip rates comparable to the learned baselines. The sampled unrelated concepts have low baseline prevalence, so this check covers a limited set of potential side effects.

D.4 Reconstruction at Matched Level-1 Sparsity

We compare SAE++ and MSAE on ImageNet with Qwen3-VL-4B-Instruct at the same Level-1 dictionary dimensionality and sparsity k_1. Fraction of Variance Unexplained (FVU) measures normalized reconstruction error; lower is better.

Table 10: Level-1 FVU at matched dictionary dimensionality and sparsity on ImageNet with Qwen3-VL-4B-Instruct.

Level-1 sparsity k_1	SAE++ FVU	MSAE FVU
3	0.006266	0.005837
6	0.004639	0.004800
12	0.003432	0.003320

The methods have comparable reconstruction quality at matched sparsity: MSAE has lower FVU at $k_1 = 3$ and 12, while SAE++ has lower FVU at $k_1 = 6$.

D.5 Empirical Training Stability

Table 11 reports training diagnostics for an ImageNet + Qwen3-VL-4B-Instruct run with TopK sparsities $k_1 = 6$ and $k_2 = 1$.

Table 11: Training diagnostics for one joint-training run. Token counts are rounded. FVE is Fraction of Variance Explained; higher is better.

Step (tokens)	Level-1 reconstruction	Level-2 reconstruction	FVE
5k (13M)	5.36×10^{-3}	4.65×10^{-3}	0.9279
10k (26M)	4.53×10^{-3}	2.48×10^{-3}	0.9390
20k (51M)	6.97×10^{-4}	2.00×10^{-4}	0.9906
40k (102M)	3.86×10^{-4}	1.13×10^{-5}	0.9948
80k (205M)	3.64×10^{-4}	1.13×10^{-5}	0.9951
160k (410M)	3.43×10^{-4}	1.07×10^{-5}	0.9954
195k (500M)	3.43×10^{-4}	1.07×10^{-5}	0.9954

Both reconstruction losses decrease during training and remain unchanged at the reported precision over the final approximately 90M tokens, indicating an empirical plateau in this run.

E Implementation Details

E.1 MLLMs, Datasets, and Implementation Details

MLLM activation extraction. We evaluate `Qwen3-VL-4B-Instruct`, `Gemma-3-4B-IT`, and `LLaVA-1.5-13B`.

For `Qwen3-VL-4B-Instruct`, we collect activations from visual transformer block 23.

For `Gemma-3-4B-IT`, we use vision tower encoder layer 26.

For `LLaVA-1.5-13B`, we extract hidden states from language backbone layer 39.

All activations are collected with the prompt "Describe the image content accurately."

Datasets. We use Color [46], ImageNet [47], iNaturalist [48], and COCO [49].

For the Color dataset, we use its mono-semantic subset, which contains 1,000 images, and adopt an 8:2 train/test split.

For ImageNet, we randomly sample 50 images from each of the 1,000 classes in the training set, obtaining 50,000 training images, and evaluate on all 50,000 images from the ImageNet validation set.

For iNaturalist, we randomly sample 5 images from each of 10,000 species for both training and testing, forming a 1:1 train/test split.

For COCO, we randomly sample 250 images from each of the 80 predefined categories for both training and testing.

Training and implementation. For all methods, we fix the number of Level-1 and Level-2 neurons to $n_1 = 20{,}000, \quad n_2 = 10{,}000$, respectively, to ensure a fair comparison.

All models are optimized using Adam, and all experiments are conducted on 4 NVIDIA RTX PRO 6000 GPUs.

Table 12: Training configuration reported for the hierarchy experiments. The downstream steering configuration is described separately below.

Item	Value
Optimizer	Adam
Learning rate	10^{-4}
Batch size	256 activation tokens
Training steps	1,953,125 (approximately 500M tokens)
Dictionary sizes	$n_1 = 20{,}000$, $n_2 = 10{,}000$
TopK sparsities	$k_1 = 6$, $k_2 = 1$
Soft sparsity penalties	$\lambda_1 = 0.1$, $\lambda_2 = 0.5$
Hardware	4 NVIDIA RTX PRO 6000 GPUs

The soft sparsity penalties apply to variants that use these penalties. The stacked baseline uses $k_1 = 6$ and $k_2 = 1$, satisfying $k_2 < k_1$. The downstream steering configuration is described separately in Appendix E.6.

For baseline methods that do not natively support learning multi-level concepts, such as BatchTopK, TopK, ReLU, P-Annealing, Gated, and JumpReLU SAEs, we train two separate SAEs: one with $n = n_1$ and one with $n = n_2$.

The resulting Level-1 and Level-2 units are connected post hoc using the parent-assignment rule in Appendix E.5.

For Matryoshka SAE, we use its native nested dictionary structure as the hierarchical representation.

For the stacked SAE baseline, we use the naive stacked formulation discussed in Sec. 4; its empirical performance is reported in Sec. 5.2.

E.2 Dynamic Masking of Dead Latents

To reduce unnecessary computation in the Level-2 SAE, we use a dynamic masking strategy for inactive Level-1 atoms.

At each training step t, let $\mathcal{B}$ denote the current mini-batch.

We define the active atom set

$$\mathcal{A}_t = \left\{ \mathbf{w}_j \;\middle|\; \sum_{\mathbf{x} \in \mathcal{B}} \mathbf{1}(|[g_1(\mathbf{x})]_j| > \epsilon_0) \geq 1 \right\}, \tag{34}$$

where $\epsilon_0 > 0$ is a small threshold and $[g_1(\mathbf{x})]_j$ denotes the j-th entry of the Level-1 sparse code.

Instead of applying the Level-2 reconstruction loss to all n_1 decoder atoms at every step, we apply it only to active atoms in $\mathcal{A}_t$:

$$\mathcal{L}_{2,t} = \frac{1}{|\mathcal{A}_t|} \sum_{\mathbf{w}_j \in \mathcal{A}_t} \Big(\|\mathbf{w}_j - h_2(g_2(\mathbf{w}_j))\|_2^2 + \lambda_2 \mathcal{S}(g_2(\mathbf{w}_j)) \Big). \tag{35}$$

If $\mathcal{A}_t = \emptyset$, we skip the Level-2 update for that mini-batch. This masking is an implementation strategy rather than a change to the model architecture. It reduces computation by avoiding repeated Level-2 updates on atoms that are inactive for the current batch. Across training, atoms are included in the Level-2 objective whenever they become active in sampled mini-batches.

E.3 Stop-Gradient Ablations for the Level-2 Objective

In our full SAE++ formulation, the Level-2 objective in Eqn. (9) is optimized without any stop-gradient operation, allowing gradients from $\mathcal{L}_2$ to flow through the Level-1 decoder atoms $\{\mathbf{w}_j\}_{j=1}^{n_1}$. To better understand the role of this gradient coupling, we conduct an ablation study with two stop-gradient variants.

Partial Stop-Gradient. In this variant, we apply a stop-gradient operator to $\mathbf{w}_j$ only in the reconstruction target of the first term in Eqn. (9). Concretely, the Level-2 loss is modified as

$$\mathcal{L}_2^{\text{partial}} = \frac{1}{n_1} \sum_{j=1}^{n_1} \Big(\|\text{sg}[\mathbf{w}_j] - h_2(g_2(\mathbf{w}_j))\|_2^2 + \lambda_2 \mathcal{S}(g_2(\mathbf{w}_j)) \Big), \tag{36}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator. This detaches the reconstruction target, while reconstruction and sparsity gradients can still reach the Level-1 atoms through the encoder input.

Bilateral Stop-Gradient. In this variant, we apply the stop-gradient operator to $\mathbf{w}_j$ for both terms in the Level-2 objective:

$$\mathcal{L}_2^{\text{full}} = \frac{1}{n_1} \sum_{j=1}^{n_1} \Big(\|\text{sg}[\mathbf{w}_j] - h_2(g_2(\text{sg}[\mathbf{w}_j]))\|_2^2 + \lambda_2 \mathcal{S}(g_2(\text{sg}[\mathbf{w}_j])) \Big). \tag{37}$$

This blocks Level-2 gradients into the Level-1 atoms; the atoms can still change through Level-1 training.

No Stop-Gradient (Full SAE++). By contrast, our full method does not employ any stop-gradient operation in Eqn. (9). This allows the Level-2 reconstruction and sparsity objectives to jointly shape the Level-1 decoder atoms, encouraging them to organize into structures that admit compact, semantically coherent higher-level abstractions. Empirically, this formulation yields the strongest performance, as reported in our ablation results.

E.4 Baselines

We compare SAE++ with a diverse set of SAE baselines that represent common dictionary-learning and mechanistic-interpretability architectures:

- **TopK SAE** [38] enforces sparsity by retaining only the top-K activations per input.
- **BatchTopK** [50] applies Top-K sparsity at the batch level, encouraging more balanced feature usage across mini-batches.
- **ReLU SAE** [13] uses a standard ReLU nonlinearity with an ℓ_1-style sparsity objective.
- **P-Annealing** [37] gradually increases sparsity during training for improved optimization stability.
- **Gated SAE** [39] introduces learnable gates to decouple feature selection from feature magnitude estimation.
- **JumpReLU** [40] uses a discontinuous activation function for sharper feature selection.
- **Matryoshka SAE** [21, 16] uses nested dictionaries to encourage multi-scale feature separation.
- **Stacked SAE** stacks a second SAE on the sparse code of the first SAE, corresponding to the naive architecture analyzed in Sec. 4.

These baselines cover hard sparsity, soft sparsity, adaptive sparsity schedules, gated feature selection, discontinuous activations, nested-prefix hierarchies, and naive stacked hierarchies.

For flat SAE baselines, parent-child structure is constructed post hoc using the rule in Appendix E.5, so that all methods can be evaluated with the same HMS metric.

E.5 Evaluation Metrics

Hierarchical Mono-Semanticity (HMS). Existing monosemanticity evaluations mainly ask whether an individual SAE feature corresponds to a coherent semantic concept [45]. In contrast, our goal is to evaluate *hierarchical* concept discovery: given a high-level Level-2 concept, do its assigned Level-1 children represent semantically related low-level concepts? We therefore define Hierarchical Mono-Semanticity (HMS), which measures the semantic coherence among the Level-1 concepts grouped under the same Level-2 parent.

Level-1 semantic representations. For each Level-1 SAE neuron j, we construct a semantic representation $\mathbf{R}_j \in \mathbb{R}^{d_e}$ from a fixed vision-backbone embedding space. Let $\{\mathbf{E}_i\}_{i=1}^N$ denote image embeddings extracted from a fixed vision backbone, where $\mathbf{E}_i \in \mathbb{R}^{d_e}$ is the embedding of image i. Let $v_{i,j}$ be the activation value of Level-1 neuron j on image i. We first min-max normalize the activations of neuron j across the dataset:

$$a_{i,j} = \frac{v_{i,j} - \min(\mathbf{v}_j)}{\max(\mathbf{v}_j) - \min(\mathbf{v}_j) + \epsilon}, \qquad \mathbf{v}_j = \{v_{i,j}\}_{i=1}^N, \tag{38}$$

where $\epsilon > 0$ is a small constant for numerical stability. The semantic representation of neuron j is then the activation-weighted average of image embeddings:

$$\mathbf{R}_j = \frac{\sum_{i=1}^N a_{i,j}\mathbf{E}_i}{\sum_{i=1}^N a_{i,j} + \epsilon}. \tag{39}$$

Intuitively, $\mathbf{R}_j$ summarizes the visual semantics of the images on which neuron j is most active. Thus, two Level-1 neurons with similar top-activating image semantics should have nearby $\mathbf{R}_j$'s in the vision-backbone embedding space.

Parent assignment. Let $\pi(j) \in \{1, \dots, n_2\}$ denote the parent assignment that maps each Level-1 neuron j to a Level-2 concept. For SAE++, this assignment is induced by the Level-2 SAE. Each Level-1 neuron j corresponds to a decoder atom

$$\mathbf{w}_j = \mathbf{W}^{(1)}_{\mathrm{dec},j} \in \mathbb{R}^d,$$

which is encoded by the Level-2 encoder g_2. We assign j to the most active Level-2 latent:

$$\pi(j) = \operatorname*{argmax}_{\ell \in \{1,\dots,n_2\}} [g_2(\mathbf{w}_j)]_\ell. \tag{40}$$

For baseline SAE variants that do not natively define a parent–child hierarchy, we construct $\pi(\cdot)$ post hoc. We train or obtain Level-1 and Level-2 units according to the corresponding baseline design, and assign each Level-1 unit to the Level-2 unit with which it has the highest co-activation frequency over the evaluation dataset. This yields a unified parent assignment for all methods, allowing HMS to be computed with the same formula.

HMS definition. For each Level-2 concept k, define its Level-1 children as

$$\mathcal{C}_k = \{j : \pi(j) = k\}, \tag{41}$$

and let $m = |\mathcal{C}_k|$. When $m < 2$, we exclude k from aggregate HMS statistics because pairwise coherence is undefined for singleton or empty child sets. For $m \geq 2$, the HMS score of concept k is the mean pairwise cosine similarity among the semantic representations of its Level-1 children:

$$\mathrm{HMS}(k) = \frac{2}{m(m-1)} \sum_{\substack{u,v \in \mathcal{C}_k \\ u<v}} \frac{\mathbf{R}_u^\top \mathbf{R}_v}{\|\mathbf{R}_u\|_2 \|\mathbf{R}_v\|_2}. \tag{42}$$

The normalization $2/(m(m-1))$ averages over all unordered child pairs. A high HMS score means that the children assigned to the same Level-2 parent are mutually close in semantic embedding space, indicating a coherent high-level grouping. A low HMS score indicates that the parent mixes semantically unrelated or weakly related Level-1 concepts.

Aggregate statistics. Let $\mathcal{K}_{\text{active}}$ be the set of Level-2 concepts with at least two alive children and nonzero semantic vectors. Our main paper reports HMS_{mean}, which summarizes the overall semantic coherence of all active Level-2 concepts:

$$\text{HMS}_{\text{mean}} = \frac{1}{|\mathcal{K}_{\text{active}}|} \sum_{k \in \mathcal{K}_{\text{active}}} \text{HMS}(k). \tag{43}$$

For completeness, the full quantitative tables in Appendix F also report:

$$\text{HMS}_{\text{min}} = \min_{k \in \mathcal{K}_{\text{active}}} \text{HMS}(k), \tag{44}$$

$$\text{HMS}_{\text{med}} = \text{median}_{k \in \mathcal{K}_{\text{active}}} \text{HMS}(k), \tag{45}$$

$$\text{HMS}_{\text{max}} = \max_{k \in \mathcal{K}_{\text{active}}} \text{HMS}(k). \tag{46}$$

The mean is used as the primary summary metric because it reflects overall hierarchy quality across all active Level-2 concepts. The median provides a robustness check for the typical parent concept, while the minimum and maximum characterize worst-case and best-case parent concepts.

E.6 Steering Details

Setup. The MLLM we use is Qwen3-VL-4B-Instruct, with both SAEs operating on the residual stream of `visual.blocks.23`. SAE++ is the two-level cascade with the number of Level-1 and Level-2 neurons to $n_1 = 20{,}000$ and $n_2 = 10{,}000$, respectively. MSAE is a global-batch-top-k Matryoshka SAE with $d = 30\text{k}$, group sizes $[20\text{k}, 10\text{k}]$, $k = 20$. Both SAEs are trained on the same $20{,}000$ COCO val2014 images.

Concept clusters and MSAE matching. We randomly select ten Level-2 SAE++ clusters. The L1 neurons with a concept word list were obtained by prompting the MLLM on the top-10 training images. MSAE has no native cluster structure, so for each SAE++ cluster we discover a matched MSAE set $\mathcal{A}_{\text{MSAE}}$ by selecting the $|\mathcal{A}|$ MSAE atoms with lowest average activation rank on the *same* training images.

Steering hook. A forward hook on `visual.blocks.23` encodes the residual activation through the SAE, overrides each neuron in $\mathcal{A}$ to a clamp value c, and writes back the SAE reconstruction. The clamp magnitude is $c = \pm 3\mathbf{u}_s$, where $\mathbf{u}_s$ is the mean post-ReLU activation of the cluster's atoms on the cluster's top-10 training images, computed independently for each SAE. The hook is identical for the two SAEs.

Methods. For every (cluster, image) pair we generate four captions: **baseline** (no clamp), **rand1** (one alive atom clamped, deterministic seed), **best_child** (the highest-scoring atom in $\mathcal{A}$), and **cluster** (every atom in $\mathcal{A}$ clamped jointly). Insertion uses $+3\mathbf{u}_s$, suppression $-3\mathbf{u}_s$.

Image sets. Insertion uses 100 random held-out images per cluster, identical between the two SAEs ($1{,}000$ rows each). Suppression filters to images where the unsteered baseline already mentions the concept (suppression success is defined only when the target is present in the unsteered caption); we collect 1000 qualifying SAE++ rows and evaluate MSAE on the exact same 1000 pairs.

Caption generation. For every (cluster, image, arm) we run one greedy generation prompted with *"Describe what is in this image in one short sentence."* (`do_sample=False`, `max_new_tokens=30`, no system prompt). The same prompt is used in every condition; only the activation hook changes.

LLM judge. For each row, the four arm captions are anonymized to labels A/B/C/D under a row-wise random permutation. Gemini 2.5 Flash is shown the test image and the four labelled captions and answers, for each label, the binary question *"does the caption semantically express the concept?"* with a one-sentence reason. Results under all four judges are reported in Table 3.

F Full Quantitative Results

Full Results on HMS Scores. Table 14$\sim$16 show the full results for all HMS scores across different MLLMs and different datasets.

Table 13: Ablation studies on ImageNet with `Qwen3-VL-4B-Instruct`.

Metric	Partial SG	Bilateral SG	Stacked	HAC	Spectral	SAE++ (Full)
$\mathbf{HMS}_{\min}$	0.0263	0.0101	**0.2765**	0.0000	0.0000	<u>0.0380</u>
$\mathbf{HMS}_{\text{med}}$	0.8280	0.7012	0.3367	<u>0.8587</u>	0.6906	**0.9999**
$\mathbf{HMS}_{\max}$	**0.9999**	**0.9999**	0.3670	0.9893	<u>0.9907</u>	**0.9999**
$\mathbf{HMS}_{\text{mean}}$	<u>0.7099</u>	0.6333	0.3322	0.7025	0.6490	**0.7776**

Ablation Studies. Table 13 evaluates which components are responsible for the gains of SAE++: (1) Restricting gradient flow between the two SAE levels hurts performance: both partial and bilateral stop-gradient variants (**Partial SG** and **Bilateral SG**, details in Appendix E.3) lead to lower HMS scores compared with full joint training (**SAE++ (Full)**), indicating that the Level-2 objective should shape the Level-1 atoms rather than merely cluster a fixed dictionary. (2) The Stacked SAE (**Stacked**) performs substantially worse, supporting our theoretical argument that naively re-compressing sparse activation codes is poorly suited for multi-level concept learning. (3) Replacing the Level-2 SAE with post-hoc clustering on Level-1 decoder weight columns $\mathbf{w}_j$, including Hierarchical Agglomerative Clustering (**HAC**) and Spectral Clustering (**SC**), also underperforms our full SAE++. This suggests that the learned Level-2 sparse abstraction is more effective than post-hoc clustering.

Table 14: HMS scores for on `Qwen3-VL-4B-Instruct`. We mark the best results in **bold** and the second best with <u>underline</u>.

Dataset	Metric	BatchTopK	TopK	ReLU	P-Annealing	Gated	JumpReLU	Matryoshka	Stacked	SAE++ (Ours)
Color	$\mathbf{HMS}_{\min}$	0.1893	0.1015	0.2048	0.3830	0.2947	0.1030	**0.9710**	0.4585	<u>0.5797</u>
	$\mathbf{HMS}_{\text{med}}$	0.4202	0.1076	0.2125	0.5451	0.5200	0.1267	<u>0.9990</u>	0.6279	**0.9999**
	$\mathbf{HMS}_{\max}$	0.8704	0.1581	0.2250	0.7073	0.9900	0.3344	<u>0.9995</u>	0.9964	**0.9999**
	$\mathbf{HMS}_{\text{mean}}$	0.4207	0.1224	0.2141	0.5450	0.6547	0.1465	<u>0.9775</u>	0.6012	**0.9825**
ImageNet	$\mathbf{HMS}_{\min}$	<u>0.1341</u>	0.1318	0.0572	0.0693	0.0311	0.0597	0.1104	**0.2765**	0.0380
	$\mathbf{HMS}_{\text{med}}$	0.1413	0.1387	0.6041	<u>0.7984</u>	0.5612	0.0680	0.6146	0.3367	**0.9999**
	$\mathbf{HMS}_{\max}$	0.6092	0.2124	<u>0.9772</u>	0.9334	0.9289	0.0990	0.9979	0.3670	**0.9999**
	$\mathbf{HMS}_{\text{mean}}$	0.2564	0.1554	0.5624	<u>0.6393</u>	0.5613	0.0756	0.5843	0.3322	**0.7776**
COCO	$\mathbf{HMS}_{\min}$	0.1116	0.1158	0.1118	<u>0.1204</u>	0.0867	0.0894	0.0334	0.1156	**0.1900**
	$\mathbf{HMS}_{\text{med}}$	0.1220	0.1436	0.1255	0.1257	0.1923	0.1171	<u>0.4598</u>	0.1468	**0.9999**
	$\mathbf{HMS}_{\max}$	0.1650	0.1864	0.9471	0.1310	0.8634	0.1554	<u>0.9978</u>	0.1731	**0.9999**
	$\mathbf{HMS}_{\text{mean}}$	0.1270	0.1514	0.3275	0.1257	0.2809	0.1206	<u>0.4614</u>	0.1443	**0.9795**
iNaturalist	$\mathbf{HMS}_{\min}$	0.0851	0.0838	**0.1225**	0.1071	0.1041	0.0867	0.0892	<u>0.1099</u>	0.0323
	$\mathbf{HMS}_{\text{med}}$	0.2436	0.2648	0.2001	0.1400	0.1928	0.1003	<u>0.5983</u>	0.2254	**0.9738**
	$\mathbf{HMS}_{\max}$	0.8308	0.9257	0.2573	**0.9999**	0.8418	0.1139	0.9931	0.2850	**0.9999**
	$\mathbf{HMS}_{\text{mean}}$	0.3182	0.3254	0.1934	0.4157	0.2611	0.1016	<u>0.5879</u>	0.2307	**0.7412**

Table 15: HMS scores for on `Gemma-3-4B-IT`. We mark the best results in **bold** and the second best with <u>underline</u>.

Dataset	Metric	BatchTopK	TopK	ReLU	P-Annealing	Gated	JumpReLU	Matryoshka	Stacked	SAE++ (Ours)
Color	$\mathbf{HMS}_{\min}$	0.4740	0.1876	0.2001	0.5925	0.1106	0.1264	<u>0.9750</u>	0.5901	**0.9814**
	$\mathbf{HMS}_{\text{med}}$	0.6412	0.3885	0.2237	0.6584	0.6123	0.1301	<u>0.9845</u>	0.6612	**0.9936**
	$\mathbf{HMS}_{\max}$	0.8850	0.9991	0.2425	0.7243	0.8146	0.2523	<u>0.9995</u>	0.9911	**0.9997**
	$\mathbf{HMS}_{\text{mean}}$	0.6667	0.4452	0.2221	0.6602	0.5662	0.1597	<u>0.9849</u>	0.7097	**0.9929**
ImageNet	$\mathbf{HMS}_{\min}$	0.1448	0.1294	0.0792	0.0125	-0.0025	0.0238	0.2010	<u>0.2975</u>	**0.4155**
	$\mathbf{HMS}_{\text{med}}$	0.1589	0.1351	0.0798	0.0899	0.0883	0.0867	<u>0.7532</u>	0.4991	**0.8375**
	$\mathbf{HMS}_{\max}$	0.1729	0.1407	0.0805	0.2312	0.7234	0.9608	<u>0.9926</u>	0.7007	**0.9998**
	$\mathbf{HMS}_{\text{mean}}$	0.1605	0.1338	0.0797	0.1501	0.2564	0.1571	<u>0.6874</u>	0.4879	**0.7750**
COCO	$\mathbf{HMS}_{\min}$	0.1930	0.0527	**0.4659**	0.1384	0.1956	0.1312	0.2187	<u>0.2205</u>	0.0293
	$\mathbf{HMS}_{\text{med}}$	0.2802	0.0530	0.4659	0.1596	<u>0.8284</u>	0.1736	0.7250	0.5083	**0.9999**
	$\mathbf{HMS}_{\max}$	0.3675	0.0533	0.4659	0.1809	0.9736	0.1952	**0.9999**	0.6769	**0.9999**
	$\mathbf{HMS}_{\text{mean}}$	0.2802	0.0530	0.4659	0.1596	0.6659	0.1667	<u>0.7020</u>	0.4124	**0.8907**
iNaturalist	$\mathbf{HMS}_{\min}$	0.2109	0.1572	0.0713	0.1333	-0.0020	0.1010	**0.7186**	0.2472	<u>0.4012</u>
	$\mathbf{HMS}_{\text{med}}$	0.5028	0.2299	0.0892	0.1481	0.0959	0.1435	<u>0.8863</u>	0.4557	**0.9607**
	$\mathbf{HMS}_{\max}$	0.7947	0.3027	0.1072	0.1629	**0.9999**	**0.9999**	0.9879	0.6642	0.9763
	$\mathbf{HMS}_{\text{mean}}$	0.5028	0.2299	0.0892	0.1481	0.2254	0.4149	<u>0.8687</u>	0.4601	**0.9128**

Table 16: HMS scores for on `LLaVA-1.5-13B`. We mark the best results in **bold** and the second best with underline.

Dataset	Metric	BatchTopK	TopK	ReLU	P-Annealing	Gated	JumpReLU	Matryoshka	Stacked	SAE++ (Ours)
Color	HMS_{min}	0.2481	0.1104	0.0587	0.1968	0.0979	0.2013	**0.9273**	0.2536	0.5672
	HMS_{med}	0.3220	0.2396	0.2475	0.3250	0.3525	0.3088	0.9377	0.6788	**0.9481**
	HMS_{max}	0.4461	0.4154	**0.9999**	0.5128	**0.9999**	**0.9999**	**0.9999**	0.9953	**0.9999**
	HMS_{mean}	0.3307	0.2421	0.3170	0.3357	0.4324	0.4337	0.9618	0.5767	**0.9705**
ImageNet	HMS_{min}	0.1043	0.0796	0.0168	0.0113	0.1414	0.0289	**0.2260**	0.1631	0.0942
	HMS_{med}	0.1751	0.2176	0.0814	0.4980	0.1976	0.2757	0.7792	0.1990	**0.9650**
	HMS_{max}	0.6983	0.9541	0.5679	0.8122	0.2811	0.9870	0.9949	0.6534	**0.9999**
	HMS_{mean}	0.1913	0.2458	0.1100	0.4934	0.1945	0.4099	0.7471	0.2304	**0.8958**
COCO	HMS_{min}	0.0459	0.0855	0.0766	0.0991	0.0895	**0.1173**	0.0482	0.1166	0.0624
	HMS_{med}	0.3296	0.3382	0.2535	0.2706	0.0895	0.2068	0.8646	0.2031	**0.9970**
	HMS_{max}	0.9918	0.9954	0.9969	0.9758	0.0895	0.5017	0.9998	0.7896	**1.0000**
	HMS_{mean}	0.3966	0.3963	0.3206	0.3339	0.0895	0.2368	0.8107	0.2843	**0.9405**
iNaturalist	HMS_{min}	0.0902	0.1322	0.0688	0.1126	0.0880	0.0059	0.1117	0.2093	**0.3531**
	HMS_{med}	0.2984	0.2865	0.3124	0.2419	0.2929	0.2122	0.7855	0.2286	**0.9871**
	HMS_{max}	0.9420	0.8361	0.9883	0.6619	0.9958	0.9956	0.9969	0.3690	**0.9992**
	HMS_{mean}	0.3330	0.3156	0.3644	0.2655	0.3633	0.2575	0.7586	0.2598	**0.7798**

G More Qualitative Results

In Fig. 5, we provide more qualitative results on our SAE++'s learned Level-2 and Level-1 concepts from MLLMs, compared with Matryoshka SAEs.



Figure 4: More qualitative results on our SAE++'s learned Level-1 and Level-2 concepts from MLLMs, compared with Matryoshka SAEs.

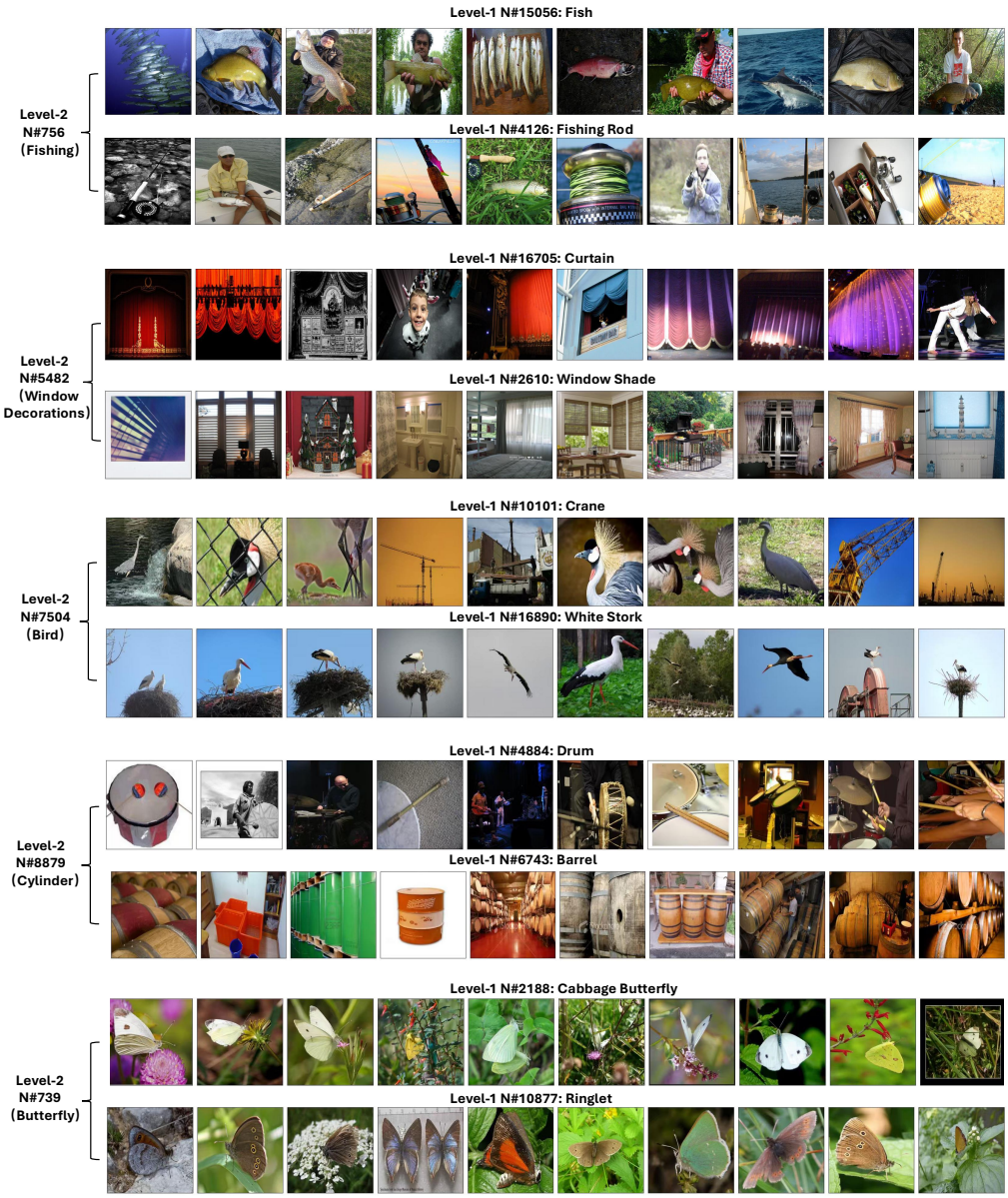


Figure 5: More qualitative results on our SAE++'s learned Level-1 and Level-2 concepts from MLLMs.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Sections 5 and 5.2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code is available at `https://github.com/Wang-ML-Lab/sae-plus-plus`. The benchmark datasets are cited in the paper, and implementation and evaluation details are provided in Appendix E.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, see Section. E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Appendix D.3.1 reports means and standard deviations across ten target concepts and paired t-tests on matched concept-level steering success rates. Appendix D.3.2 reports means and standard deviations across twenty unrelated concepts for the off-target evaluation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: Our model used public LLMs and datasets, and we can not foresee any potential societal impact.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We did it properly.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The SAE++ repository provides installation, dataset access, training, evaluation, and checkpoint usage instructions, together with the MIT code license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The study does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are used to evaluate target-concept presence in steered captions and off-target concept presence. Their roles and evaluation protocols are described in Appendices E.6 and D.3.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.