Do Sparse Autoencoders Learn Meaningful Concept Hierarchies?

Nils Grandien[1,*] **David Steinmann**[1,2,*] **Felix Friedrich**[3]
Kristian Kersting[1,2,4]
[1]Computer Science Department, TU Darmstadt [2]Hessian Center for AI (hessian.AI)
[3]Black Forest Labs [4]German Research Center for AI (DFKI)

Abstract

Sparse autoencoders (SAEs) have become an important tool for unsupervised concept discovery in large models. To make the resulting feature spaces more interpretable and manageable, recent approaches have begun imposing hierarchical structure, either explicitly or as an implicit effect of training constraints, yet rigorous comparison remains difficult. There are no agreed-upon requirements for what a meaningful feature hierarchy should satisfy, and evaluation has largely relied on qualitative illustrations with fragmented quantitative protocols. To address this, we derive a set of key requirements for generalization/specialization hierarchies in unsupervised concept discovery, drawing on semantic net and taxonomy research alongside recent SAE work, and use them to derive a concrete evaluation protocol. Applying this protocol to current SAE approaches trained on visual data, we find that while feature spaces generally provide a basis for sensible hierarchies, establishing good hierarchical structure remains challenging. In particular, feature absorption, both in its well-known hard form and in a continuous, soft form, systematically compromises hierarchy quality, pointing to a fundamental tension that future approaches will need to navigate.[1]

1 Introduction

A central challenge in deep learning is balancing model expressiveness with interpretability and user control. Concept-based representations have emerged as a structured middle ground, supporting inherent explanation [26, 27, 42] and direct model control [31, 52]. Unsupervised concept discovery approaches, such as sparse autoencoders (SAEs), recover such representations without predefined labels [6, 12, 17, 38, 45, 48].

As these representations grow larger and more general, however, navigating and interacting with them becomes increasingly difficult [43], a limitation that stems in part from their lack of meaningful structure (cf. Fig. 1). Humans naturally organize concepts hierarchically [32, 34], increasing the interest in hierarchies within this context for their clear semantics and natural abstraction capabilities [7, 41, 49]. A number of methods now aim to learn such hierarchical structures within SAE representation spaces [2, 7, 11, 29, 38, 58]. Yet each relies on its own criteria and evaluation protocol, making meaningful comparison difficult and leaving the field without shared objectives for what a hierarchy in this setting should achieve.

In this work, we derive a set of general requirements for *generalization/specialization* hierarchies in the context of unsupervised concept learning, drawing on established ideas from taxonomy evaluation [50] and semantic nets [5] alongside recently introduced metrics from the SAE literature [33].

*These authors share equal contribution.
[1]The code for this work is available at: `https://github.com/nlsgrndn/eval-HSAEs`

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

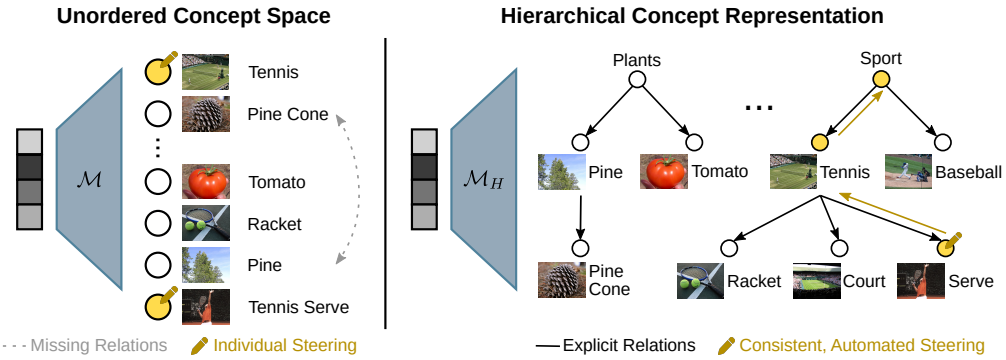


Figure 1: **Hierarchical concept representations explicitly model relations between concepts and allow for consistent model steering.** Typical unsupervised concept discovery approaches (left) provide large and unordered concept spaces. These lack clear connections between concepts, making interpretation difficult and cumbersome. Similarly, model steering requires identifying and updating many concepts at once. Hierarchical concept representations (right) instead explicitly provide a hierarchy graph for interpretation as well as consistent and automated model steering.

Importantly, we combine static, structural requirements, such as *hierarchical abstractness*, with dynamic, co-activation-based requirements, such as *frequent refinement*, the latter being particularly important in the context of unsupervised concept discovery. From these requirements, we derive a concrete set of metrics to assess the hierarchical capabilities of SAE-based concept extractors, which we use to evaluate current approaches that impose hierarchical structure either explicitly or implicitly during training. In doing so, we identify the current state of the field and concrete guidelines for developing future hierarchical concept extractors.

In summary, our contributions are (1) establishing a general set of requirements for *generalization/specialization* hierarchies in unsupervised concept learning, (2) deriving concrete metrics to evaluate concept extractors against these requirements, (3) evaluating current SAE approaches trained on visual data on the quality of their hierarchical structure, and (4) identifying concrete guidelines for the future development of hierarchical concept extractors.

This work is structured as follows: We discuss related work in Sec. 2, before introducing our requirements and corresponding metrics in Sec. 3. We provide an experimental analysis of previous SAE approaches in Sec. 4, discuss main implications in Sec. 5, before concluding the paper in Sec. 6.

2 Related Work

Concept Hierarchies. Concept hierarchies can be understood as a specific type of (directed, acyclic) knowledge graph [1] in which relations carry a unified semantics [5]. Common instantiations are *is-a* hierarchies [5] and part-whole hierarchies [25], both of which impose a clear generalization/specialization ordering on concepts. A long line of work has studied the automated construction of such hierarchies, also known as taxonomies, from text, ranging from lexico-syntactic patterns [21] to recent LLM-based approaches [54, 59], and has developed corresponding evaluation frameworks. These range from qualitative criteria such as conciseness and comprehensiveness [36, 53] to quantitative supervised benchmarks [4], and unsupervised measures such as information richness [50] or consistency between taxonomic distance and embedding similarity [56]. In unsupervised settings, concept hierarchies are related to hierarchical clustering approaches, with metrics such as cophenetic correlation [16] or the cost metric by Dasgupta [13]. For unsupervised representation spaces, Park et al. [39] investigate the representation of hierarchical concepts directly and provide a perspective on the geometry of hierarchical concept representations. Our work builds on this, but shifts the focus to hierarchies in the context of unsupervised concept discovery independent of representation geometry. This setting introduces an additional dimension absent from classical taxonomy evaluation: beyond static, structural properties, the hierarchy is instantiated dynamically by a concept encoder, raising the question of how concepts co-activate and refine one another on concrete inputs.

Structure in SAEs. SAEs decompose model activations into a high-dimensional set of sparse latent features, and have become a widely used framework for unsupervised concept discovery in large language and vision-language models [6]. Several post-hoc approaches investigate what structure these feature spaces contain, for example, via decoder geometry [6, 28, 30], top-activating inputs [18], textual descriptions [23], or co-activation patterns [7, 57]. Wittenmayer et al. [55] explicitly target a hierarchical DAG, using weighted conditional feature probabilities. Beyond post-hoc analysis, several approaches impose structural constraints during training. Zaigrajew et al. [58] and Bussmann et al. [7] follow the Matryoshka principle using nested sparsity constraints and nested dictionary sizes, respectively, while Li and Ren [29] introduce a non-nested static grouping strategy. Costa et al. [11] instead recover hierarchical features by iteratively refining reconstruction residuals, Muchane et al. [35] enforce hierarchy explicitly by combining a high-level SAE with lower-level expert SAEs, and Luo et al. [33] learn multiple SAEs and their relations via structural constraints and feature perturbations. Not all of these works frame their contribution in hierarchical terms, but each imposes implicit or explicit structure that shapes relationships among learned features. This work introduces a shared framework to assess and compare how well these approaches impose meaningful hierarchies in SAE latent spaces.

SAE Evaluation. Standard SAE evaluation focuses on reconstruction quality, sparsity, and feature disentanglement [24], with more recent work also targeting the interpretability of individual features via automated tools [40] or Monosemanticity scores [20, 38]. However, the evaluation of hierarchical structure has received limited attention. Supervised approaches compare against ground-truth hierarchies, typically on small synthetic datasets [7, 8, 9, 11]. Among unsupervised approaches, Ye et al. [57] evaluate hierarchical strength via parent-child versus child-child co-occurrence ratios, Luo et al. [33] measure hierarchical consistency and conditional co-activation probabilities, and Wittenmayer et al. [55] assess extracted feature graphs through a user study; each covering different aspects in isolation. Moreover, existing work relies heavily on qualitative illustrations [7, 35, 55], making it difficult to compare methods rigorously. Rather than adding another individual metric, we first establish an explicit, comprehensive set of requirements for what a meaningful hierarchy should satisfy, and then derive a unified, quantitative evaluation protocol from those requirements.

3 Concept Hierarchies

We consider a concept as "the label of a set of things that have something in common" [3], for example, a *tennis racket* or a *pine cone*, but also more abstract ideas such as *plant*. When reasoning about visual scenes, humans naturally relate concepts and organize them into structures [19], with hierarchies being a particularly natural choice due to their clear relational semantics. In the following, we formalize this notion and derive the requirements that such hierarchies should satisfy in the context of unsupervised concept learning.

Hierarchical Semantics Let $\mathcal{F}$ be a set of concepts used to represent inputs $x \in \mathcal{X}$. We are interested in *generalization/specialization* hierarchies over $\mathcal{F}$, which have a clear semantic relation: for a parent concept $p \in \mathcal{F}$ and a child concept $c \in \mathcal{F}$, p represents a more general property of inputs than c while c captures a more specific property within the scope of p. Consequently, the presence of c typically implies the presence of p [5]. This includes typical IS-A relations (tennis is a sport) but also more general relationships (a tennis racket is a specific aspect of tennis).

We model these relations as a **concept hierarchy graph**, a directed acyclic graph (DAG), defined as $G := (V, E)$ over a subset $\tilde{\mathcal{F}} \subseteq \mathcal{F}$ of all concepts that are part of at least one hierarchical relation, i.e. $V := \tilde{\mathcal{F}}$. For an edge $(p, c) \in E$, p is the parent and c is the child. We assume that G is transitively reduced, meaning an edge (p, c) only exists if there is no directed path of length ≥ 2 from p to c, which avoids redundant ancestry information and keeps the local structure interpretable. The DAG allows concepts with multiple parents, better reflecting the semantics of generalization/specialization hierarchies. A **concept extractor** $\mathcal{M}$ maps inputs to a set of feature activations $a(x) = \{a_f(x) \in \mathbb{R}_{\geq 0} | f \in \mathcal{F}'\}$ in its feature space $\mathcal{F}'$. We assume that there is a mapping from $\mathcal{F}'$ to $\mathcal{F}$, and drop this difference to simplify the notation in the following. We also treat $a(x)$ as covering the case where $\mathcal{M}$ operates on latent embeddings $e(x)$ of an encoder model, as in the SAE setting.

We now describe the requirements for concept hierarchies from the perspective of unsupervised concept learning, in particular, requirements for hierarchical concept extractors.[2] We write $\mathcal{M}_H$ for a hierarchical concept extractor, p for a parent concept, c for a child concept, and $C(p)$ for the set of children of p in the concept hierarchy graph.

3.1 Requirements

In this section, we derive requirements that capture the essential properties of concept hierarchies, divided into concept-level, structure-level, and co-activation-level, depending on whether they concern concepts in isolation, the static graph structure, or the dynamic instantiation of that structure across individual samples.

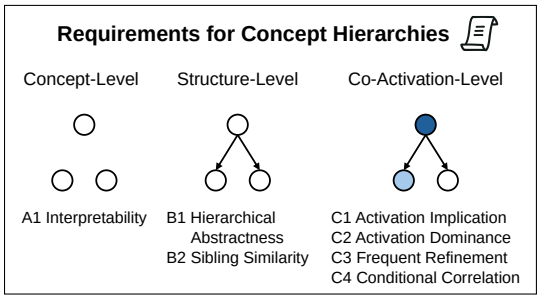


Figure 2: **Requirements for concept hierarchies operate on different graph aspects.** *Concept-level* concerns individual features, *structure-level* addresses the static graph structure, and *co-activation-level* evaluates the hierarchy given sample-based activations.

Concept-level. Concept-level requirements concern individual concepts in isolation, independent of their position in the hierarchy graph. They represent basic pre-requisites for meaningful hierarchy analysis: studying hierarchical relations is only worthwhile if the underlying concepts satisfy basic quality criteria.

Requirement A1: Interpretability. Each node in the graph should correspond to a human-understandable concept from the data $\mathcal{X}$. This is particularly relevant in unsupervised concept discovery, where there is no supervision to guarantee the quality of individual features.

Structure-level. These requirements describe desired properties of the hierarchy graph independent of individual samples. Assuming concepts are interpretable, these requirements specify how the position of concepts in G should align with their semantics.

Requirement B1: Hierarchical Abstractness. The semantics of a *generalization/specialization* hierarchy specify that for an edge $(p, c) \in E$, the child c should be more specific than its parent p. For example, *tennis* is more general than *tennis racket* or *tennis serve*. This requirement is either explicitly [5] or implicitly [33, 35, 55] assumed in prior work.

Requirement B2: Sibling Similarity. If a parent p has multiple children, each child should share the concept represented by p, forming a coherent set of refinements rather than an arbitrary collection. In our example, all children of *tennis* should be semantically related through their shared parent concept.

Co-activation-level. The third level of requirements goes beyond the static hierarchy and considers sample-based activations. In unsupervised concept discovery, it is not sufficient for the hierarchy to encode meaningful semantic relations; the per-sample activation behavior should also reflect the semantics of parent-child relations. We treat activations as indicators of the strength of evidence for a concept and consider both discrete (present or absent) and continuous (larger activations equal stronger evidence) settings. Importantly, this level also addresses the problem of absorption [8].

Requirement C1: Activation Implication. Since c is a refinement within the scope of p, the presence of c implies the presence of p: an active child should always be accompanied by an active parent. This is a standard assumption in hierarchical sparse coding [22] and a central property of *generalization/specialization* hierarchies [5]. Activation Implication also means that there is no hard form of absorption, i.e., the activation of a child concept does not suppress the activation of its parent concept. In our example, every input where *tennis racket* is active, *tennis* should also be active.

Requirement C2: Activation Dominance. When both p and c are active, the activation of the parent should be at least as strong as that of the more specific child. For example, evidence for *tennis serve* should be accompanied by equal or stronger evidence for *tennis*. This need not hold as a hard

[2]This assumes a method to obtain a concept hierarchy graph from the concept extractor, either provided inherently [35] or constructed post-hoc [55].

per-sample constraint: it is possible to be confident about a *racket* without knowing whether it is for *tennis* or *squash*, but it should hold in expectation over the full data distribution. This requirement is strongly related to *activation implication*, refining the activation semantics for an edge $(p, c) \in E$ where both p and c are active.

Requirement C3: Frequent Refinement. When an abstract parent p is present in an input, it is often not present in its abstract form, but rather in the form of one of its child refinements. This should also be reflected in the co-activations: an active parent p should frequently be accompanied by at least one active child $c \in C(p)$. While refinement frequency is desired to be high, it is not a universal constraint: a parent may appear without an active child if the relevant specialization is not contained in $C(p)$ or if the parent is present in a generic form that needs no further refinement. In our example, an image containing a *tennis racket* should activate not only *tennis* but also *tennis racket*.

Requirement C4: Conditional Correlation. While *activation dominance* concerns only the binary comparison of parent-child pairs, this requirement takes a broader view: the activation magnitude of p should *correlate* positively with the activation magnitudes of its active children in $C(p)$, given that at least one child is active. This directly addresses a "soft" form of absorption, in which stronger child activations gradually suppress the parent rather than triggering a hard activation drop. In our example, across all samples where at least one child of *tennis* is active, the activation strength of *tennis* should correlate with that of its active children, regardless of which children are active in any given sample.

3.2 Metrics

We now provide concrete metrics to measure each requirement. We assume a concept extractor with a corresponding concept hierarchy graph, but do not assume access to a ground-truth hierarchy. Therefore, all metrics evaluate requirements in an unsupervised manner. Depending on the requirement, metrics operate at the feature level, the parent-child level, or the parent-child group level (cf. Tab. 9).

Notation. Given a dataset D, we write $\mathbb{P}_d[\cdot]$ for the empirical probability over D. Given a (hierarchical) concept extractor $\mathcal{M}_H$, each feature $f \in V$ induces an activation $a_f(x) \in \mathbb{R}_{\geq 0}$, with its binarized form $A_f(x) \in \{0, 1\}$ which indicate whether the feature f is considered *active* on input x. We write $\text{TopKInputs}_f := \{x^{(1)}, \ldots, x^{(k)}\}$ for the k inputs in D with the largest activations under a_f, and $\text{sim}(u, v) := \frac{\langle u,v \rangle}{\|u\| \, \|v\|}$ for cosine similarity.

Monosemanticity Score $\mathtt{s}_m$. Concept-level interpretability has been extensively studied in the SAE literature, with a range of metrics and evaluation protocols available [40]; we therefore do not introduce a dedicated interpretability metric here, but refer to related work for such comparisons [24, 47]. Instead, we use the Monosemanticity Score, following Pach et al. [38], which evaluates semantic coherence in an unsupervised manner and provides an important baseline perspective for the hierarchical metrics below. Concretely, it computes the average pairwise similarity between input embeddings, weighted by feature activations. Let $\tilde{a}_f(x)$ be the min-max normalized activations of f on x, and $e(x)$ the latent embeddings of x from an input encoder, then $\mathtt{s}_m \in [-1, 1]$ is

$$\mathtt{s}_m(f) := \frac{\sum_{x^{(i)}, x^{(j)} \in D} \; \tilde{a}_f(x^{(i)}) \; \tilde{a}_f(x^{(j)}) \; \text{sim}\big(e(x^{(i)}), e(x^{(j)})\big)}{\sum_{x^{(i)}, x^{(j)} \in D} \; \tilde{a}_f(x^{(i)}) \; \tilde{a}_f(x^{(j)})} \,. \tag{1}$$

Hierarchical Abstractness Score $\mathtt{s}_{ha}$. This metric is computed per parent-child pair and uses the Monosemanticity Score as a proxy for specificity, treating specificity as the inverse of abstractness: a high $\mathtt{s}_m$ indicates a more specific feature, while a low suggests a more abstract concept. Let $(p, c) \in E$ be a parent-child edge in the graph and $\mathbf{1}[\cdot]$ the indicator function. We define the binary

$$\mathtt{s}_{ha}(p, c) := \mathbf{1}[\mathtt{s}_m(p) < \mathtt{s}_m(c)] \,. \tag{2}$$

Sibling Similarity Score $\mathtt{s}_{ss}$. This metric measures the similarity among the children $c \in C(p)$ of a parent p, using cosine similarity between activation-weighted averages of input encoder latents, inspired by Dreyer et al. [15]. For a parent $p \in V$, the $\mathtt{s}_{ss} \in [-1, 1]$ is defined as:

$$\mathtt{s}_{ss}(p) := \frac{2}{|\mathcal{C}(p)| \, (|\mathcal{C}(p)| - 1)} \sum_{c_i, c_j \in \mathcal{C}(p); i<j} \text{sim}\big(\bar{e}_{f_i}, \bar{e}_{f_j}\big) \;, \tag{3}$$

with $\bar{e}_f := \frac{\sum_{x \in \text{TopKInputs}_f} a_f(x) \, e(x)}{\sum_{x \in \text{TopKInputs}_f} a_f(x)}$. This metric is only defined for $|\mathcal{C}(p)| \geq 2$.

Activation Implication Score $\mathtt{s}_{ai}$. We measure how often a parent is active given an active child via the conditional activation probability on binarized activations, similar to [33]. For $(p, c) \in E$:

$$\mathtt{s}_{ai}(p, c) := \mathbb{P}_D\big[A_p(x) = 1 \,\big|\, A_c(x) = 1\big]. \tag{4}$$

Activation Dominance Score $\mathtt{s}_{ad}$. This metric measures the fraction of inputs on which the parent activation exceeds the child activation, conditioned on both features being active. Conditioning on the child being active prevents the metric from being dominated by how often a child is active relative to its parent. Conditioning on the parent being active excludes cases that are already captured by *activation implication*, i.e., where the parent is inactive while the child is active. For $(p, c) \in E$:

$$\mathtt{s}_{ad}(p, c) := \mathbb{P}_D[a_p(x) \geq a_c(x) \,|\, (A_p(x) = 1) \wedge (A_c(x) = 1)]\,. \tag{5}$$

Refinement Frequency Score $\mathtt{s}_{rf}$. This metric measures how often at least one of a parent's children is active, given that the parent is itself active. As we do not assume specific dependencies between siblings, we consider only the presence of at least one active child rather than the total number of active children. While it is desirable that a parent is often accompanied by its children, this does not always have to be the case, as discussed for the respective requirement. For $p \in V$:

$$\mathtt{s}_{rf}(p) := \mathbb{P}_D\left[\sum\nolimits_{c \in \mathcal{C}(p)} A_c(x) \geq 1 \,\middle|\, A_p(x) = 1\right]. \tag{6}$$

Conditional Correlation Score $\mathtt{s}_{cc}$. This metric measures the Spearman correlation ρ_s between activations of a parent feature and the maximum activation of any of its children. It is conditioned on at least one child being active to measure correlations among active features, rather than how often a child is active relative to the parent. We use Spearman rather than Pearson correlation as we are interested in a monotonic rather than strictly linear relationship. For $p \in V$, the $\mathtt{s}_{cc} \in [-1, 1]$ is:

$$\mathtt{s}_{cc}(p) := \rho_s\Big(\{a_p(x)\}_{x \in D_p},\, \{\max_{c \in \mathcal{C}(p)} a_c(x)\}_{x \in D_p}\Big)\,, \tag{7}$$

with $D_p := \Big\{x \in D : \sum_{c \in \mathcal{C}(p)} A_c(x) \geq 1\Big\}$.

Metric Aggregation. Graph-level scores for each metric are obtained by averaging local scores. Thereby, parents contribute proportionally to the number of their children for parent-child metrics, whereas each parent contributes equally for parent-child-group metrics.

4 Experimental Evaluation

In this section, we evaluate the current state of SAEs with respect to their hierarchical feature representations, comparing standard TopK SAEs against several hierarchical approaches. We aim to answer the following research questions: (1) What is the current state of hierarchical visual concept learning in SAEs? (2) How does the choice of vision encoder affect hierarchy quality? (3) How does hierarchy quality vary with depth? Let us first describe the experimental setup.

SAE Approaches. We compare six SAE types: a standard TopK SAE [6], a Matryoshka SAE with nested sparsity constraints (ActMSAE) [58], a Matryoshka SAE with nested dictionary sizes (ArchM-SAE) [7], EWG-SAE [29], Matching-Pursuit SAE (MP-SAE) [11], and H-SAE [35]. Hyperparameter details for each method are provided in Sec. B.3.

Data and Encoders. All SAE approaches are trained on global image embeddings from CLIP ViT-L/14 [44], DINOv2-base [37] as well as patch-level residual stream encodings of Gemma-4 E2B [51] over CC3M [46]. The dataset is divided into three disjoint splits: graph creation (500,000 samples), evaluation (500,000 samples), and the rest for SAE training.

Graph Creation. Evaluating our metrics requires a concrete concept hierarchy graph. Since most hierarchical SAE approaches do not produce an explicit graph but only impose implicit structural constraints during training, we follow Wittenmayer et al. [55] and construct a DAG from pairwise conditional activation probabilities between features. We apply several post-processing steps to the resulting graph: removing noise by discarding overly small components and overly dense nodes, followed by a transitive reduction to eliminate redundant edges. Further details are given in Sec. B.4.

Table 1: **SAEs learn mostly shallow concept hierarchies.** The graph-level statistics for SAEs trained on CLIP highlight different graph sizes and densities, while all methods except ActMSAE are mostly limited to two-level hierarchies. Values are mean ± std across three seeds.
†Group Size Deviation is the standard deviation of the per-seed mean group size across seeds.

Model	Nodes	Edges	Parent-Child Groups	Max Depth	Avg. Group Size	Group Size Deviation†	Nodes at Depth ≥ 3
TopK SAE	2120 ± 42	3011 ± 77	225 ± 16	3.67 ± 0.58	13.40 ± 0.92	19.03 ± 1.64	80 ± 15
EWGSAE	974 ± 57	2641 ± 192	168 ± 10	2.67 ± 0.58	15.77 ± 1.09	22.14 ± 2.04	3 ± 3
ArchMSAE	1012 ± 99	1683 ± 190	99 ± 2	3.00 ± 0.00	16.98 ± 1.61	19.16 ± 6.37	2 ± 1
ActMSAE	3341 ± 19	6170 ± 81	1372 ± 14	5.67 ± 0.58	4.50 ± 0.09	8.99 ± 1.07	1147 ± 29
H-SAE	6137 ± 0	5776 ± 0	361 ± 0	2.00 ± 0.00	16.00 ± 0.00	0.00 ± 0.00	0 ± 0

Evaluating Learned Concept Hierarchies. We first compare the general graph structures across methods (cf. Tab. 1 for CLIP, Sec. C.1 for further details). Post-hoc constructed graphs cover up to half of all SAE features, with MP-SAE being the only exception: its training strategy reduces co-activation of similar features (Sec. B.4), resulting in an empty graph. The explicit hierarchy of H-SAE, by contrast, always contains all 6,137 features. Notably, ActMSAE is the only method producing a hierarchy with more than two levels, suggesting better co-activation across multiple levels of abstraction. Graph statistics are stable across seeds, indicating that the learned structures are not sensitive to the SAE training process.

Turning to the graph metrics (cf. Fig. 3), several clear trends emerge. The baseline TopK SAE already produces features that can form a meaningful concept hierarchy: the extracted graph covers roughly one-third of all features and achieves strong abstractness and sibling similarity scores. However, its co-activation metrics reveal that absorption remains a key challenge. Even when hard absorption is controlled for via the graph construction threshold, TopK SAE shows low $\mathtt{s}_{ad}$ and $\mathtt{s}_{cc}$, meaning parent and child activations are largely inconsistent in strength. The low $\mathtt{s}_{rf}$ further confirms that, while features are semantically grouped, children rarely co-activate with their parents.

EWG-SAE shows no meaningful improvement over the baseline and often scores slightly worse. The Matryoshka approaches perform better on co-activation metrics overall, with ActMSAE generally exceeding ArchMSAE. ArchMSAE predominantly learns abstract features with no clear hierarchical abstraction (as indicated by around random $\mathtt{s}_m$ and $\mathtt{s}_{ha}$), suggesting that nested dictionary sizes reduce absorption but constrain co-activations to rather abstract features. ActMSAE, with its nested sparsity constraints, instead produces more frequent and consistent co-activations of specific features with their general parents, as reflected in its high $\mathtt{s}_{rf}$ and multi-level graph.

The explicit hierarchy of H-SAE yields consistent co-activation behavior and high refinement frequency by design, but soft absorption persists: parent and child activation strengths remain poorly correlated. Moreover, its features achieve near-random $\mathtt{s}_m$ and near-zero $\mathtt{s}_{ha}$, indicating that the imposed hierarchy lacks semantic meaning, as children are more abstract than their parents or contain no clear interpretation (cf. Sec. C.6). Thus, explicit structural constraints can enforce co-activation consistency, but do not reduce the necessity to maintain clear semantic structure within the hierarchy.

Comparing Hierarchies Across Vision Encoders. To investigate the impact of the encoder on hierarchical feature learning, we compare SAEs trained on CLIP embeddings with those trained on DINOv2 and Gemma-4 E2B. In contrast to CLIP's contrastive vision-language pretraining, DINO is trained self-supervised on vision data alone, and for Gemma-4, we train SAEs on the residual stream of patch-level image encodings. Differences between encoders include the substantially lower average similarity between image embeddings for DINOv2 (≈ 0.01), which results in smaller graphs on average, and that MP-SAE produces a non-empty graph when trained on Gemma-4 E2B, albeit consisting only of one parent-child group (cf. Tab. 11). The otherwise stable trends across encoders are a strong indicator that, as long as the embeddings contain enough information to learn suitable hierarchies, the challenges of learning hierarchical concept spaces are driven by SAE training strategies rather than the underlying embedding space.

Investigating More Than Two-Level Hierarchies. Since all methods except ActMSAE produce shallow two-level hierarchies, we use ActMSAE on CLIP to investigate how hierarchy quality changes with depth (cf. Fig. 4). The $\mathtt{s}_m$ and $\mathtt{s}_{ha}$ values reveal a clear difference in feature specificity between levels 1 and 2, while deeper levels remain similar. Refinement frequency increases with

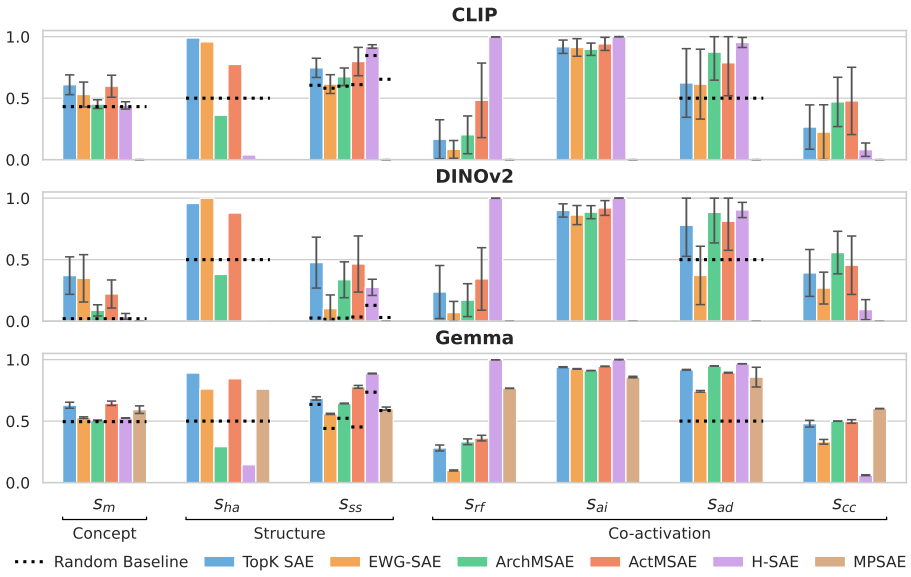


Figure 3: **Graph metrics for all SAEs trained on CLIP, DINOv2, and Gemma-4 encodings.** Average and standard deviation over nodes, edges, or parent-child groups. Random baseline uses a random selection of respective samples/nodes/edges where applicable. Full data distributions (cf. Sec. C.6) show that values < 0 can occur for $\mathtt{s}_{cc}$. As $\mathtt{s}_{ha}$ is binary, we omit its standard deviation.

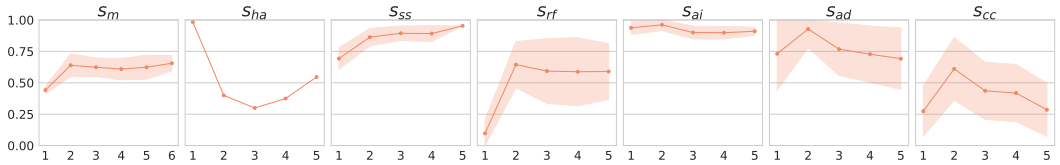


Figure 4: **Hierarchy quality of ActMSAE across depths.** While the hierarchy abstraction diminishes at higher depth, semantic coherence among children and refinement frequency increase. Values are aggregated at the depth level of the node ($\mathtt{s}_m$) or of the parent for edge and group metrics.

depth: top-level abstract features rarely co-activate with their children, while more specific features co-occur much more frequently. The remaining co-activation metrics show no clear trend, suggesting that absorption behavior is largely independent of hierarchy depth. Overall, ActMSAE produces a deeper graph with a meaningful abstraction gap between the first two levels, and lower levels consist of frequently co-occurring features with increasingly subtle hierarchical relations.

Ablations on Graph Creation. Since the results for most SAE types also depend on the method used to build the hierarchy graph, we ablate the graph-creation threshold, the graph post-processing, and the similarity measure used for graph creation. Varying the threshold on conditional activations (Fig. 5) mainly affects $\mathtt{s}_{ai}$, $\mathtt{s}_{cc}$, and graph size, which drops noticeably at very high thresholds, but the relative comparison between SAE methods remains stable. The post-processing steps improve human readability (transitive reduction) or hierarchy quality (graph pruning) without altering the main findings (cf. Sec. C.2). Finally, replacing conditional co-activations (CCA) with masked cosine similarity (MCS) preserves the main trends, though MCS-based graphs show lower $\mathtt{s}_{ai}$, $\mathtt{s}_{ad}$, and $\mathtt{s}_{cc}$. MCS does allow constructing a graph for MPSAE for CLIP, but this graph has near-random $\mathtt{s}_m$ and near-zero $\mathtt{s}_{ha}$, indicating that its hierarchy lacks semantic meaning (cf. Tab. 15 in the appendix).

Qualitative Hierarchy Analysis. We complement the quantitative evaluation with qualitative examples of learned hierarchies. The first (cf. Fig. 6, left) shows a dense hierarchy grouping various bird types and their specializations. Interestingly, angels are also grouped as birds, likely due to their wings and feathers. Chicken are further refined via roosters, hens and eggs, which highlights a sensible more than two-level categorization. On the other hand, this also illustrates that child

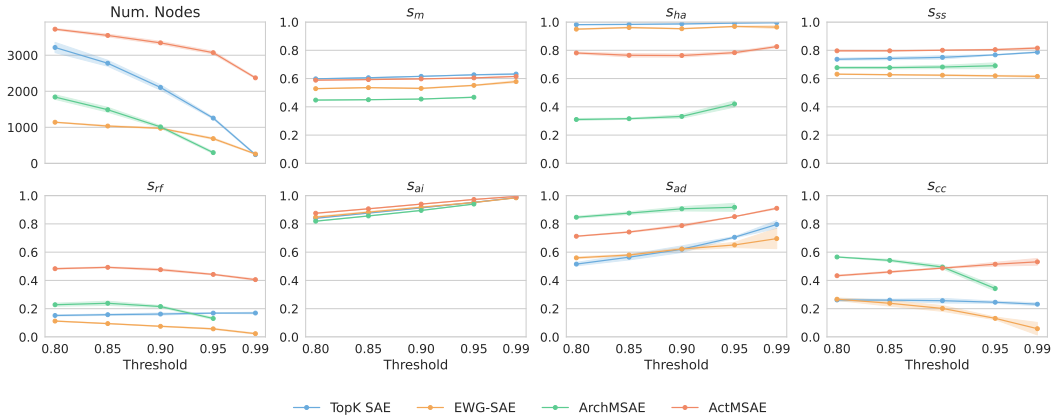


Figure 5: **Ablation of the graph-creation threshold on CLIP.** Mean $\pm$ standard deviation over three seeds. While the threshold on conditional co-activation affects graph size and the co-activation-level metrics, the relative comparisons between SAE types remain largely consistent.

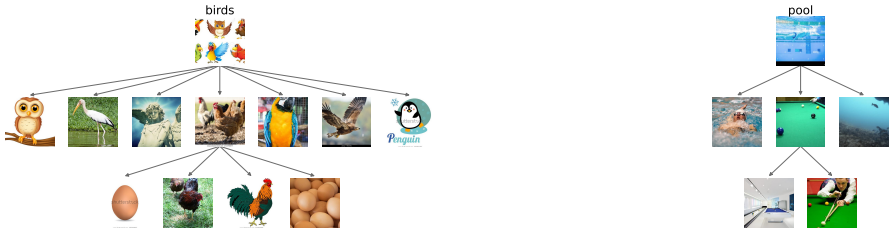


Figure 6: **Learned hierarchies capture meaningful semantic abstractions.** Left: various bird types are grouped under a common *birds* concept, with chickens further refined into roosters, hens and eggs. Right: CLIP's vision-language pretraining shapes the learned hierarchies, causing swimming pools and billiards tables to be grouped under the shared concept *pool*. All examples taken from ActMSAE trained on CLIP.

activation may sometimes exceed parent activation in a sensible way. For samples with an egg, it can very well be that the model is more certain about the egg than about whether it is an egg of a chicken, highlighting why the activation dominance and conditional correlation metrics should be viewed in expectation over the dataset instead of hard per-sample constraint.

The second example illustrates a more interesting case: the top-level concept *pool* is refined into swimming pools, fishing ponds, and billiards, with billiards further split into two subcategories. This grouping reflects CLIP's language supervision, which places visually distinct concepts close together due to the shared word *pool*. This highlights how the choice of encoder can directly shape the hierarchical connections that emerge, a factor quantitative metrics alone do not fully capture. On the other hand, there are two typical failure modes of learned concept hierarchies: grouping only spuriously co-activating features and the presence of noisy features that lack a consistent underlying concept (cf. Sec. C.6 for concrete examples of these failure modes).

5 Discussion

Our evaluations show that SAE feature spaces generally contain features that can be grouped into meaningful hierarchies, but obtaining consistent activation behavior remains difficult. A central source of this difficulty is feature absorption, which is typically discussed in its "hard" form, i.e., whether a feature is active or not [8]. Our evaluation reveals a complementary "soft" form: when both parent and child are active, stronger child activations tend to suppress parent activations, as captured by *activation dominance* and *conditional correlation*. Matryoshka approaches reduce hard absorption, but remain susceptible to this soft form. Addressing both forms of absorption might require rethinking the sparsity constraints central to current SAE training. A less sparse SAE with consistent hierarchical

activations could be more useful than a sparser one with no meaningful structure. A second open question is whether hierarchical structure should be explicitly enforced during training or implicitly induced and recovered post-hoc. Explicit constraints can guarantee that certain requirements are met, while post-hoc graph construction adapts more flexibly to the features the SAE has learned. A promising direction likely combines both: fixing a high-level hierarchical structure with clear activation semantics, while discovering the lower parts of the hierarchy flexibly.

In general, our metrics matter differently depending on the use case. For understanding the learned concept hierarchy itself (exploring the concept space), co-activation metrics are less important while structure- and concept-level metrics are the focus. For steering, relevance depends on the scenario: if the target concept and its parent are already known to be semantically sensible, co-activation metrics matter most. If this is not yet established, structure- and concept-level metrics are needed first to verify the hierarchy is meaningful. More generally, these metrics build on each other: trusting the structure-level metrics assumes that the concept-level metrics are reasonable, and trusting the co-activation metrics assumes that the structure-level metrics are reasonable in turn.

Limitations. By design, all metrics in this work operate unsupervised, approximating, for example, semantic similarity through image embeddings rather than human judgment. While this enables scalable evaluation, and we verify the structure-level metrics against the known, ground-truth ImageNet-1k hierarchy [14] in Tab. 16, this set of metrics does not fully capture whether a hierarchy is meaningful to humans. Thus, quantitative metrics and human inspection are best understood as complementary tools rather than substitutes for one another. A second limitation concerns the dependency on the graph construction method of Wittenmayer et al. [55] for all approaches except H-SAE. Since this method considers only conditional activation probabilities, it captures just one of the hierarchy requirements and may not yield the best possible graph for every SAE. More broadly, how to best extract a hierarchy graph from SAE latent representations remains an open question that warrants further investigation.

6 Conclusion

In this work, we derived a general set of requirements for concept hierarchies for unsupervised concept learning, covering concept-level prerequisites alongside structure- and co-activation-level requirements. From these, we developed a concrete set of metrics for evaluating SAE-based concept hierarchies. Our evaluation of current hierarchical SAE approaches shows that they can discover features that can be grouped into semantically meaningful, although mostly shallow, hierarchies. However, activation behavior remains inconsistent across hierarchy levels, and absorption emerges as a central challenge, both in its previously known hard form and in a newly identified soft form, where stronger child activations suppress parent activations even when both are active. Looking ahead, the proposed requirements and metrics can serve as direct training constraints for future hierarchical concept extractors. Open questions remain how to best extract hierarchy graphs from SAE latent spaces, and how much hierarchical structure should be explicitly enforced during training. Applying this evaluation protocol to other modalities or concept extractors beyond SAEs is a natural next step towards a better understanding of unsupervised hierarchical concept learning.

Acknowledgements

This work was supported by the "ML2MT" project from the Volkswagen Stiftung, by the German Research Foundation (DFG) under Germany's Excellence Strategy (EXC 3066/1 "The Adaptive Mind", Project No. 533717223 and by the EU-funded "TANGO" project (EU Horizon 2023, GA No 57100431). It has benefited from the HMWK project Hessian.AI, and from the Cluster of Excellence "Reasonable AI" funded by the DFG under Germany's Excellence Strategy EXC-3057.

References

[1] Bilal Abu-Salih. Domain-specific knowledge graphs: A survey. *Journal of Network and Computer Applications*, 185:103076, 2021.

[2] Dana Arad, Aaron Mueller, and Yonatan Belinkov. Saes are good for steering–if you select the right features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10252–10270, 2025.

[3] E James Archer. The psychological nature of concepts. In *Analyses of concept learning*, pages 37–49. Elsevier, 1966.

[4] Georgeta Bordea, Els Lefever, and Paul Buitelaar. Semeval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *Proceedings of the 10th international workshop on semantic evaluation (semeval)*, pages 1081–1091, 2016.

[5] Ronald J. Brachman. What is-a is and isn't: An analysis of taxonomic links in semantic networks. *Computer;(United States)*, 10, 1983.

[6] 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. URL `https://transformer-circuits.pub/2023/monosemantic-features/index.html`.

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

[8] David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, Satvik Golechha, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint arXiv:2409.14507*, 2024.

[9] David Chanin, Tomáš Dulka, and Adrià Garriga-Alonso. Feature hedging: Correlated features break narrow sparse autoencoders. *arXiv preprint arXiv:2505.11756*, 2025.

[10] Luciano da F Costa. Further generalizations of the jaccard index. *arXiv preprint arXiv:2110.09619*, 2021.

[11] Valérie Costa, Thomas Fel, Ekdeep Singh Lubana, Bahareh Tolooshams, and Demba Ba. From flat to hierarchical: Extracting sparse representations with matching pursuit. *arXiv preprint arXiv:2506.03093*, 2025.

[12] Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

[13] Sanjoy Dasgupta. A cost function for similarity-based hierarchical clustering. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 118–127, 2016.

[14] 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. Ieee, 2009.

[15] Maximilian Dreyer, Jim Berend, Tobias Labarta, Johanna Vielhaben, Thomas Wiegand, Sebastian Lapuschkin, and Wojciech Samek. Mechanistic understanding and validation of large ai models with semanticlens. *Nature Machine Intelligence*, 7(9):1572–1585, 2025.

[16] James S Farris. On the cophenetic correlation coefficient. *Systematic Zoology*, 18(3):279–285, 1969.

[17] Thomas Fel, Victor Boutin, Louis Béthune, Rémi Cadène, Mazda Moayeri, Léo Andéol, Mathieu Chalvidal, and Thomas Serre. A holistic approach to unifying automatic concept extraction and concept importance estimation. *Advances in Neural Information Processing Systems*, 36:54805–54818, 2023.

[18] Leander Girrbach and Zeynep Akata. Sparse autoencoders are topic models. *arXiv preprint arXiv:2511.16309*, 2025.

[19] Alon Hafri and Chaz Firestone. The perception of relations. *Trends in Cognitive Sciences*, 25 (6):475–492, 2021.

[20] Ruben Härle, Felix Friedrich, Manuel Brack, Stephan Wäldchen, Björn Deiseroth, Patrick Schramowski, and Kristian Kersting. Measuring and guiding monosemanticity. *Advances in Neural Information Processing Systems (NeurIPS)*, 38, 2025.

[21] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 volume 2: The 14th international conference on computational linguistics*, 1992.

[22] Rodolphe Jenatton, Julien Mairal, Francis R Bach, and Guillaume R Obozinski. Proximal methods for sparse hierarchical dictionary learning. In *International Conference on Machine Learning (ICML)*, pages 487–494, 2010.

[23] Nick Jiang, Xiaoqing Sun, Lisa Dunlap, Lewis Smith, and Neel Nanda. Interpretable embeddings with sparse autoencoders: A data analysis toolkit. *arXiv preprint arXiv:2512.10092*, 2025.

[24] Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum McDougall, Kola Ayonrinde, et al. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability. *arXiv preprint arXiv:2503.09532*, 2025.

[25] C Maria Keet and Alessandro Artale. Representing and reasoning over a taxonomy of part–whole relations. *Applied Ontology*, 3(1-2):91–110, 2008.

[26] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International Conference on Machine Learning (ICML)*, pages 2668–2677. PMLR, 2018.

[27] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning (ICML)*, pages 5338–5348. PMLR, 2020.

[28] Patrick Leask, Bart Bussmann, Michael Pearce, Joseph Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. Sparse autoencoders do not find canonical units of analysis. *arXiv preprint arXiv:2502.04878*, 2025.

[29] Ed Li and Junyu Ren. Unlocking hierarchical concept discovery in language models through geometric regularization. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025.

[30] Yuxiao Li, Eric J Michaud, David D Baek, Joshua Engels, Xiaoqing Sun, and Max Tegmark. The geometry of concepts: Sparse autoencoder feature structure. *Entropy*, 27(4):344, 2025.

[31] Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, Feiyu Xiong, et al. Controllable text generation for large language models: A survey. *arXiv preprint arXiv:2408.12599*, 2024.

[32] Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.

[33] Yifan Luo, Yang Zhan, Jiedong Jiang, Tianyang Liu, Mingrui Wu, Zhennan Zhou, and Bin Dong. From atoms to trees: Building a structured feature forest with hierarchical sparse autoencoders. *arXiv preprint arXiv:2602.11881*, 2026.

[34] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

[35] Mark Muchane, Sean Richardson, Kiho Park, and Victor Veitch. Incorporating hierarchical semantics in sparse autoencoder architectures. *arXiv preprint arXiv:2506.01197*, 2025.

[36] Robert C Nickerson, Upkar Varshney, and Jan Muntermann. A method for taxonomy development and its application in information systems. *European journal of information systems*, 22 (3):336–359, 2013.

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[38] Mateusz Pach, Shyamgopal Karthik, Quentin Bouniot, Serge Belongie, and Zeynep Akata. Sparse autoencoders learn monosemantic features in vision-language models. *arXiv preprint arXiv:2504.02821*, 2025.

[39] Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *International Conference on Learning Representations*, volume 2025, pages 76441–76463, 2025.

[40] Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models. *arXiv preprint arXiv:2410.13928*, 2024.

[41] Federico Pittino, Vesna Dimitrievska, and Rudolf Heer. Hierarchical concept bottleneck models for vision and their application to explainable fine classification and tracking. *Engineering Applications of Artificial Intelligence*, 118:105674, 2023.

[42] Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. Concept-based explainable artificial intelligence: A survey. *ACM Computing Surveys*, 2023.

[43] Forough Poursabzi-Sangdeh, Daniel G Goldstein, Jake M Hofman, Jennifer Wortman Wortman Vaughan, and Hanna Wallach. Manipulating and measuring model interpretability. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–52, 2021.

[44] 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 (ICML)*, pages 8748–8763. PmLR, 2021.

[45] Sukrut Rao, Sweta Mahajan, Moritz Böhle, and Bernt Schiele. Discover-then-name: Task-agnostic concept bottlenecks via automated concept discovery. In *European Conference on Computer Vision (ECCV)*, pages 444–461. Springer, 2024.

[46] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018.

[47] Dong Shu, Xuansheng Wu, Haiyan Zhao, Daking Rai, Ziyu Yao, Ninghao Liu, and Mengnan Du. A survey on sparse autoencoders: Interpreting the internal mechanisms of large language models. *arXiv preprint arXiv:2503.05613*, 2025.

[48] Wolfgang Stammer, Antonia Wüst, David Steinmann, and Kristian Kersting. Neural concept binder. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:71792–71830, 2024.

[49] Ao Sun, Yuanyuan Yuan, Pingchuan Ma, and Shuai Wang. Eliminating information leakage in hard concept bottleneck models with supervised, hierarchical concept learning. *arXiv preprint arXiv:2402.05945*, 2024.

[50] Samir Tartir, I Budak Arpinar, and Amit P Sheth. Ontological evaluation and validation. In *Theory and applications of ontology: Computer applications*, pages 115–130. Springer, 2010.

[51] Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

[52] Stefano Teso, Öznur Alkan, Wolfgang Stammer, and Elizabeth Daly. Leveraging explanations in interactive machine learning: An overview. *Frontiers in Artificial Intelligence*, 6:1066049, 2023.

[53] Michael Unterkalmsteiner and Waleed Abdeen. A compendium and evaluation of taxonomy quality attributes. *Expert Systems*, 40(1):e13098, 2023.

[54] Binh Vu, Rashmi Govindraju Naik, Bao Khanh Nguyen, Sina Mehraeen, and Matthias Hemmje. Automated taxonomy construction using large language models: A comparative study of fine-tuning and prompt engineering. *Eng*, 6(11):283, 2025.

[55] Kai Wittenmayer, Sukrut Rao, Amin Parchami-Araghi, Bernt Schiele, and Jonas Fischer. Cfm: Language-aligned concept foundation model for vision. *arXiv preprint arXiv:2601.13798*, 2026.

[56] Pascal Wullschleger, Majid Zarharan, Donnacha Daly, Marc Pouly, and Jennifer Foster. No gold standard, no problem: Reference-free evaluation of taxonomies. *arXiv preprint arXiv:2505.11470*, 2025.

[57] Christine Ye, Charles O'Neill, John F Wu, and Kartheik G Iyer. Sparse autoencoders for dense text embeddings reveal hierarchical feature sub-structure. In *NeurIPS 2024 Workshop on Scientific Methods for Understanding Deep Learning*, 2024.

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

[59] Qingkai Zeng, Yuyang Bai, Zhaoxuan Tan, Shangbin Feng, Zhenwen Liang, Zhihan Zhang, and Meng Jiang. Chain-of-layer: Iteratively prompting large language models for taxonomy induction from limited examples. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 3093–3102, 2024.

A Supplementary material

A.1 Broader Impact

This work addresses the current state of hierarchical representations in unsupervised concept learning. By establishing general requirements for concept hierarchies and concrete metrics for evaluating hierarchical SAE approaches, we aim to support the development of more interpretable and structured unsupervised concept representations. We note, however, that the metrics introduced here are not a substitute for careful human inspection of discovered concepts — they are best understood as a complementary tool that can guide and inform, but not replace, human judgment when working with unsupervised concept discovery.

A.2 Compute Ressources

The experiments in this work have been run on an Nvidia RTX A6000 GPU with 48GB VRAM. Training the different SAE approaches took approximately 65 GPU-hours. The evaluation of the SAEs based on the hierarchy metrics took around 225 GPU-hours, with the largest amount of compute used for the Monosemanticity Score $\mathtt{s}_m$, which requires computing pairwise similarities between image embeddings across the dataset.

B Details on Training and Experimental Setup

B.1 Gemma-4 Implementation details

We train the SAEs on the residual stream of the frozen Gemma-4 E2B multimodal decoder [51], read at layer 26 of 35 at image patch-token positions ($d = 1536$). Each image is pushed through the vision encoder and connector with no text prompt, and we keep the hidden states at the 280 image-token positions. Gemma-4's image processor treats its soft-token budget as an upper bound and produces a per-image grid shape, so we first map every image onto a fixed 960×672 canvas (resize to cover, then center crop), which yields a fixed $14 \times 20 = 280$ soft-token grid. Therefore, the SAEs trained on Gemma-4 produce feature activations at the patch level rather than a global-image embedding like CLIP and DINOv2. Because each image is represented by 280 patches, we use 50000 images each for SAE training and graph construction, and 70000 for hierarchy evaluation. All SAEs have 6144 latents, use $k = 64$ where the architecture is top-k, and are trained for 10 epochs with three seeds.

B.2 Standard SAE Evaluation Metrics

We report standard evaluation metrics for SAEs in Tab. 2, Tab. 3 and Tab. 4. Following Zaigrajew et al. [58], we report fraction of variance unexplained (FVU)[3], cosine similarity, and L_0. Both FVU and cosine similarity are used to assess reconstruction quality. Note that L_0 is effectively constant as we use TopK with $k = 64$ for all SAEs to calculate these metrics (as all activations within TopK are greater than 0, which is usually the case). The scores are calculated on the validation split of the CC3M [46] dataset.

B.3 SAE Implementation Details

Embedding preprocessing Foundation-model embeddings were mean-centered and normalized following the preprocessing used in Zaigrajew et al. [58].

SAE Hyperparameters For TopK SAE, ActMSAE, ArchMSAE, EWG-SAE, the encoder subtracts the decoder bias b_d from the input before encoding, uses an encoder bias b_e, and adds b_d back after reconstruction (following [58]):

$$f = \sigma(W_{\mathrm{e}}(x - b_{\mathrm{d}}) + b_{\mathrm{e}})\,, \qquad \hat{x} = W_{\mathrm{d}} f + b_{\mathrm{d}}.$$

Only MP-SAE does not use b_{e} due to its iterative reconstruction procedure. H-SAE does not use any bias terms, i.e., $b_d = 0$ and $b_e = 0$ following [35].

[3] FVU is computed as $\frac{1}{N}\sum_{i=1}^{N} \frac{\mathrm{Var}(x^{(i)} - \hat{x}^{(i)})}{\mathrm{Var}(x^{(i)})}$, where N is the number of datapoints.

Table 2: Base SAE Metrics for CLIP

	FVU	Cosine Similarity	L_0
TopK SAE	0.1104 ± 0.0009	0.9441 ± 0.0003	64 ± 0.0000
EWG-SAE	0.1362 ± 0.0004	0.9291 ± 0.0002	64 ± 0.0000
ArchMSAE	0.1071 ± 0.0003	0.9448 ± 0.0002	64 ± 0.0000
ActMSAE	0.1120 ± 0.0001	0.9421 ± 0.0001	64 ± 0.0000
MP-SAE	0.0609 ± 0.0000	0.9690 ± 0.0000	64 ± 0.0000
H-SAE	0.1145 ± 0.0001	0.9407 ± 0.0001	64 ± 0.0000

Table 3: Base SAE Metrics for DINO

	FVU	Cosine Similarity	L_0
TopK SAE	0.2029 ± 0.0007	0.8950 ± 0.0002	64 ± 0.0000
EWG-SAE	0.3037 ± 0.0002	0.8342 ± 0.0002	64 ± 0.0000
ArchMSAE	0.2217 ± 0.0002	0.8823 ± 0.0001	64 ± 0.0000
ActMSAE	0.2320 ± 0.0003	0.8771 ± 0.0002	64 ± 0.0000
MP-SAE	0.1418 ± 0.0002	0.9263 ± 0.0001	64 ± 0.0000
H-SAE	0.2647 ± 0.0003	0.8560 ± 0.0002	64 ± 0.0000

All SAE variants operate on foundation-model embeddings of dimension 768 (1536 for Gemma-4). To make the models comparable, we use a latent width of 6144 for all models except H-SAE, which has 6137 total features due to its expert-based architecture. Unless stated otherwise, encoder and decoder weights are untied, the activation function is ReLU, and models with an explicit fixed sparsity constraint use $L_0 = 64$. MP-SAE is the only model with tied weights and does not use an activation function. H-SAE uses LeakyReLU instead of ReLU. Table 5 lists the model-specific architectural hyperparameters.

Implementation-specific remarks. For TopK SAE and ArchMSAE, we did not use an auxiliary loss term to prevent dead features, which is sometimes used for these architectures. For MP-SAE, we used a fixed number of matching-pursuit steps, $T = 64$, during training rather than selecting T based on a reconstruction-quality threshold. For EWG-SAE, we renormalize the group sparsity weights to sum to one. This differs from the original EWG-SAE formulation, but only changes the parameterization of the sparsity penalty.

Training hyperparameters. TopK SAE, ArchMSAE, ActMSAE, and EWG-SAE follow the training approach of [58] (Tab. 6). H-SAE was trained following Muchane et al. [35], which use a different learning rate schedule (Tab. 7).

B.4 Graph creation method

To obtain a concrete concept hierarchy graph for an SAE, we first compute feature activations over our graph-creation split of the CC3M dataset. We use these activations in the graph-creation method of Wittenmayer et al. [55][4]. Since conditional probabilities involving rarely active features are unreliable, we discard features that activate on fewer than 20 of the 500,000 samples in the graph-creation split. As a consistent (binarized) co-activation of parents and children is essential for a well-structured hierarchy, we use a threshold of $t = 0.9$ to obtain an initial graph G, which is a good balance between consistent activations on the one hand and sufficiently flexible selection to obtain larger graphs on the other (cf. Fig. 5).

Afterward, we perform several post-processing steps to denoise $G = (V, E)$ and align it with the requirements for a feature hierarchy graph. First, we prune features that only belong to small, isolated parts of the hierarchy, which are usually noisy and not semantically meaningful. Let $R \subseteq V$ denote the set of root nodes of the current graph, that is, the nodes with no incoming edges. For each

[4]They use $C_{ij} = \frac{\sum_{x \in D:\, A_i(x) \wedge A_j(x)} a_j(x)}{\sum_{x \in D:\, A_j(x)} a_j(x)}$. C_{ij} is the conditional probability of feature i being active given that feature j is active, weighted by the continuous activations of feature j on the relevant samples.

Table 4: Base SAE Metrics for Gemma-4

	FVU	Cosine Similarity	L_0
TopK SAE	0.1920 ± 0.0002	0.8985 ± 0.0001	64 ± 0.0000
EWG-SAE	0.1850 ± 0.0001	0.9048 ± 0.0001	64 ± 0.0000
ArchMSAE	0.2153 ± 0.0004	0.8853 ± 0.0003	64 ± 0.0000
ActMSAE	0.2053 ± 0.0001	0.8910 ± 0.0000	64 ± 0.0000
MP-SAE	0.1450 ± 0.0025	0.9244 ± 0.0013	64 ± 0.0000
H-SAE	0.2168 ± 0.0003	0.9452 ± 0.0001	64 ± 0.0000

Table 5: Model-specific architectural hyperparameters.

Model	Hyperparameters
TopK SAE	TopK with $K = 64$.
EWG-SAE	Non-nested group sizes: $[64, 128, 256, 512, 1024, 2048, 2112]$. Global sparsity weight: 1.0. Group sparsity weights proportional to $[0.5^7, 0.5^6, 0.5^5, 0.5^4, 0.5^3, 0.5^2, 0.5^1]$. Group sparsity weights renormalized to sum to 1.
ArchMSAE	TopK with $K = 64$. Nested widths: $M = \{96, 288, 672, 1440, 2976, 6144\}$.
ActMSAE	TopK levels: $k_1, \ldots, k_8 = \{64, 128, 256, 512, 1024, 2048, 4096, 6144\}$. Loss weights: $\alpha_1, \ldots, \alpha_8 = \{8, 7, 6, 5, 4, 3, 2, 1\}$.
MP-SAE	Tied weights. $T = 64$ matching-pursuit steps. No activation function.
H-SAE	Upper-level TopK with $k = 32$. Lower-level TopK with $k = 1$ per selected expert. 361 experts. 16 atoms per subspace. Subspace dimension: 16. Activation function: LeakyReLU. ℓ_1 penalty: 10^{-3}. Orthogonality penalty: 10^{-1}.

$r \in R$, let $\mathrm{Desc}(r)$ denote the set of descendants reachable from r, excluding r itself. We keep every root r with $|\mathrm{Desc}(r)| \geq d_{\min}$ together with all nodes in $\mathrm{Desc}(r)$, and remove all remaining nodes, with $d_{\min} = 5$. We also prune all nodes with an excessively large number of children (which often correspond to noisy, always active nodes without real semantic meaning). For each node $p \in V$, we remove p if $|C(p)| \geq t$, and set t to 5% of the total features in the SAE latent space.

Lastly, we compute the transitive reduction of the graph. That is, we remove every edge $(u, v) \in E$ for which there already exists a directed path of length at least 2 from u to v. This removes redundant edges from the graph. For example, if $p \to c$ and $c \to g$, an additional edge $p \to g$ does not change reachability but clutters the hierarchy.

Graph creation for MP-SAEs. When using the graph creation method for MP-SAEs, the result is normally an empty graph (or an extremely small graph when the thresholds for conditional activations are set very low). This is most likely because MP-SAEs are trained differently from other SAE approaches, particularly through the iterative reconstruction procedure. It reconstructs the input iteratively, using the remaining residual step by step, whereas other approaches reconstruct in a single pass. This can reduce co-activation of similar features (regarding the decoder column direction) and prevent hierarchical feature co-activation behavior, which is the basis for graph creation.

B.5 Further details for Metrics

A list of all requirements, their respective metrics, the graph elements they consider and the target space of the metrics are provided in Tab. 9. For sibling similarity that uses $\mathrm{TopKInputs}_f$, we set

Table 6: Training hyperparameters for models trained (except H-SAE)

Hyperparameter	Value
Batch size	1024
Epochs	30
Learning rate	1×10^{-4} (except TopKSAE: 5×10^{-4})

Table 7: Training hyperparameters for H-SAE

Hyperparameter	Value
Epochs	100
Batch size	32512
Peak learning rate	5×10^{-4}
Initial learning rate	1×10^{-11}
Gradient norm clipping	0.75
Warmup steps	200

$k = 20$. This choice balances a sufficiently large input set for stable estimates while ensuring that, for all features, the selected top-k inputs still correspond to sufficiently strong feature activations.

Post-TopK Activations We apply a TopK feature selection function with $k = 64$ to the activations before further processing. For most of the selected SAEs, this matches the TopK sparsity constraint used during training. ActMSAE is a partial exception, as it is trained with multiple TopK levels rather than a single fixed value of k. Moreover, even for TopK SAE and ArchMSAE, the raw feature activations can contain positive entries beyond the k features that would be retained by the TopK sparsity operator when applied with the same k as during training. However, it is unclear whether these additional activations should be interpreted as meaningful. Applying TopK during evaluation therefore both aligns the activations more closely with the training setup and enforces a common sparsity level across all SAEs, improving comparability.

Binarization Both graph construction and the evaluation metrics require binarized activations. We binarize the post-TopK activations using zero as the threshold. For most inputs across all considered SAEs, this is effectively equivalent to setting the selected top-k features to 1 and all remaining features to 0, as the selected features typically have positive activations.

Random Baseline References In the evaluations of our metrics, we report a random reference baseline for several values (cf. Fig. 3). For $\mathtt{s}_m$, the random reference is represented by the average similarity between two random image encodings. This is the $\mathtt{s}_m$ score if the activations of all inputs were the same. For $\mathtt{s}_{ha}$, the random baseline is 0.5, which is the average if children and parents have random $\mathtt{s}_m$ scores. For $\mathtt{s}_{ss}$, the reference values differ because of the activation behavior of SAEs. The reference is the average feature similarity of the SAE, measured as the cosine similarity between their weighted average embeddings. As this depends on the SAE, the concrete scores vary per SAE, as shown in Tab. 8. For $\mathtt{s}_{ad}$, the reference is again 0.5, which represents the average of a random distribution of child and parent activations.

Table 8: Sibling similarity reference values

	TopK SAE	EWG-SAE	ArchMSAE	ActMSAE	MP-SAE	H-SAE
CLIP	0.6051	0.5812	0.5976	0.6102	0.6537	0.8468
DINOv2	0.0255	0.0155	0.0247	0.0338	0.0299	0.1286
Gemma-4	0.6358	0.4404	0.5230	0.4521	0.5860	0.7352

Table 9: Overview of requirements, corresponding metrics, the graph elements on which the metrics operate, and their value ranges.

Requirement	Metric	Graph Element	Range
Feature-level			
Consistency	Monosemanticity Score $\mathtt{s}_m$	Concept	$[-1, 1]$
Structure-level			
Hierarchical Abstractness	Hierarchical Abstractness Score $\mathtt{s}_{ha}$	Parent–child	$\{0, 1\}$
Sibling Similarity	Sibling Similarity Score $\mathtt{s}_{ss}$	Parent–child group	$[-1, 1]$
Co-activation-level			
Activation Implication	Activation Implication Score $\mathtt{s}_{ai}$	Parent–child	$[0, 1]$
Activation Dominance	Activation Dominance Score $\mathtt{s}_{ad}$	Parent–child	$[0, 1]$
Frequent Refinement	Refinement Frequency Score $\mathtt{s}_{rf}$	Parent–child group	$[0, 1]$
Conditional Correlation	Conditional Correlation Score $\mathtt{s}_{cc}$	Parent–child group	$[-1, 1]$

Table 10: **Graph-level statistics for DINOv2.** Values are mean $\pm$ std across three seeds.
†Group Size Deviation is the standard deviation of the per-seed mean group size across seeds.

Model	Nodes	Edges	Parent-Child Groups	Max Depth	Avg. Group Size	Group Size Deviation†	Nodes at Depth ≥ 3
TopKSAE	833 ± 41	1075 ± 95	124 ± 10	4.67 ± 0.58	8.64 ± 0.16	13.68 ± 0.85	102 ± 14
ArchMSAE	428 ± 43	546 ± 78	51 ± 6	2.00 ± 0.00	10.62 ± 0.74	10.78 ± 0.54	0 ± 0
ActMSAE	1976 ± 10	3162 ± 84	586 ± 19	6.33 ± 0.58	5.39 ± 0.06	11.97 ± 0.29	315 ± 24
H-SAE	6137 ± 0	5776 ± 0	361 ± 0	2.00 ± 0.00	16.00 ± 0.00	0.00 ± 0.00	0 ± 0
EWGSAE	465 ± 20	1038 ± 68	91 ± 6	2.67 ± 0.58	11.44 ± 0.15	7.70 ± 0.40	1 ± 1

C Further Experimental Evaluations

C.1 Further Graph-level Statistics

In the following, we provide further details on the graph statistics for all SAE methods. In addition to the graph-level statistics for CLIP reported in the main paper (Tab. 1), we provide the corresponding overview for DINOv2 in Tab. 10 and for Gemma-4 E2B in Tab. 11. The results show similar trends across all three encoders. However, the post-hoc constructed graphs for DINOv2 are generally smaller than the CLIP or Gemma-4 graphs, suggesting that the lower base similarity between DINOv2 embeddings and their differently structured encoding space reduces the number of frequently co-activating feature pairs. On two of three seeds, MP-SAE creates a non-empty graph consisting of one parent-child group. Although this is a non-empty graph for MP-SAE, it remains a very restricted, small hierarchy.

Tab. 12, Tab. 13 and Tab. 14 report the number of nodes per hierarchy level for each method. The single node at depth 0 represents an artificially added root node that connects all other components but has no semantic significance. For all methods, the majority of nodes reside at level 1, as expected, with only TopK SAE and ActMSAE having nodes at depth ≥ 3. The number of nodes at depth 2 for H-SAE is substantially larger than for any other method, even compared to ActMSAE, which has a similar number of parent nodes at depth 1. This suggests that while the rigid structure of H-SAE enforces consistent co-activations and refinement, it does not align well with the features the SAE naturally learns, which undermines the semantic coherence of nodes at level 2.

C.2 Ablations of the Graph Creation Method

While our evaluations measure the quality of the concept hierarchy graph from all SAE methods, the results for most SAE types also depend on the underlying method to create the hierarchy graph. Next, we ablate the impact of the graph-creation threshold (Fig. 5 in the main paper), graph post-processing, and the metric used to create the graph on the insights provided in the main paper. All following results show mean and standard deviation over three seeds for SAEs evaluated on CLIP. We do not include H-SAE, as its graph is independent of post-hoc graph creation.

Table 11: **Graph-level statistics for Gemma-4 E2B.** Values are mean ± std across three seeds.
†Group Size Deviation is the standard deviation of the per-seed mean group size across seeds.

Model	Nodes	Edges	Parent-Child Groups	Max Depth	Avg. Group Size	Group Size Deviation†	Nodes at Depth ≥ 3
TopK SAE	854 ± 144	1423 ± 204	25 ± 3	3.00 ± 0.00	58.06 ± 8.14	87.86 ± 8.45	28 ± 41
ArchMSAE	639 ± 50	1389 ± 137	52 ± 4	3.00 ± 0.00	26.76 ± 2.76	28.38 ± 5.74	14 ± 9
ActMSAE	2905 ± 76	6347 ± 303	525 ± 16	4.67 ± 0.58	12.08 ± 0.25	26.78 ± 2.48	588 ± 128
H-SAE	6137 ± 0	5776 ± 0	361 ± 0	2.00 ± 0.00	16.00 ± 0.00	0.00 ± 0.00	0 ± 0
MPSAE	245 ± 216	244 ± 215	1 ± 1	1.33 ± 1.15	122.00 ± 107.53	23.67 ± 28.99	0 ± 0
EWG-SAE	1096 ± 69	2358 ± 173	107 ± 2	3.00 ± 0.00	21.98 ± 1.73	35.41 ± 7.42	6 ± 1

Table 12: **Number of nodes at each depth level for hierarchies of SAEs trained on CLIP**. Values are mean ± std across three seeds. Missing depth levels are shown as '–'.

Configuration	D0	D1	D2	D3	D4	D5	D6
TopK SAE	1 ± 0	161 ± 11	1880 ± 51	79 ± 14	1 ± 1	–	–
EWG-SAE	1 ± 0	139 ± 4	833 ± 51	3 ± 3	–	–	–
ArchMSAE	1 ± 0	84 ± 4	926 ± 99	2 ± 1	–	–	–
ActMSAE	1 ± 0	371 ± 19	1823 ± 62	949 ± 28	172 ± 19	22 ± 9	4 ± 3
H-SAE	1 ± 0	361 ± 0	5776 ± 0	–	–	–	–

Impact of the graph post-processing. As detailed in Sec. B.4, we introduce a post-processing step of the initial graphs to improve the quality of these graphs. Next, we show the impact of each post-processing step on hierarchy quality in Fig. 7. Transitive reduction affects only ActMSAE, the only method with more than two hierarchy levels, and it leaves the hierarchy metrics essentially unchanged, so it removes less-interpretable connections at no cost to hierarchy quality. The post-processing pruning step of removing nodes with too many children mainly affects TopKSAE, and omitting it enlarges the graph but lowers $\mathtt{s}_m$ and $\mathtt{s}_{rf}$. Pruning small components from the hierarchies mainly affects TopKSAE and EWGSAE, and omitting it likewise enlarges the graph while lowering $\mathtt{s}_m$, $\mathtt{s}_{rf}$ and $\mathtt{s}_{ad}$. Overall, these steps improve human-readability (transitive reduction) and hierarchy quality (the two pruning steps) without affecting the main findings.

Impact of the similarity metric. The graph creation uses conditional co-activations (CCA) as metric to determine whether there is a connection between two nodes. This controls for $\mathtt{s}_{ai}$ by design. In Tab. 15, we compare the quality of the resulting graphs when we use masked cosine similarity (MCS) [6] instead of CCA. This changes the resulting graphs but preserves the main trends seen with CCA. As expected, $\mathtt{s}_{ai}$ is substantially lower under MCS, since CCA explicitly controls for

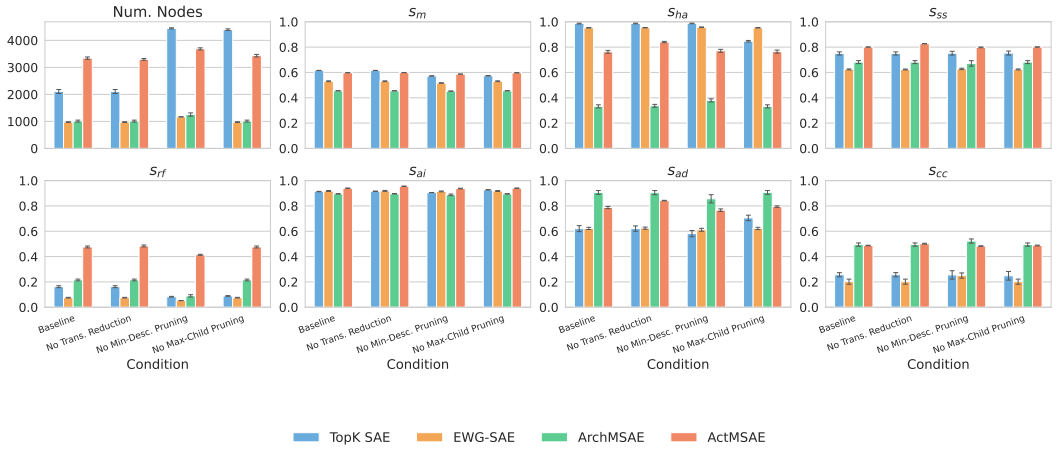


Figure 7: **Ablation of the graph post-processing on CLIP.** Mean ± standard deviation over three seeds. Post-processing steps improve hierarchy readability and quality, while trends between SAE types remain consistent.

Table 13: **Number of nodes at each depth level for hierarchies of SAEs trained on DINOv2**. Values are mean ± std across three seeds. Missing depth levels are shown as '–'.

Configuration	D0	D1	D2	D3	D4	D5	D6	D7
TopK SAE	1 ± 0	78 ± 6	653 ± 42	67 ± 15	30 ± 21	6 ± 9	–	–
EWG-SAE	1 ± 0	82 ± 4	382 ± 17	1 ± 1	–	–	–	–
ArchMSAE	1 ± 0	45 ± 5	383 ± 40	–	–	–	–	–
ActMSAE	1 ± 0	209 ± 11	1451 ± 28	254 ± 11	44 ± 9	14 ± 5	2 ± 1	0 ± 1
H-SAE	1 ± 0	361 ± 0	5776 ± 0	–	–	–	–	–

Table 14: **Number of nodes at each depth level for hierarchies of SAEs trained on Gemma-4 E2B**. Values are mean ± std across three seeds. Missing depth levels are shown as '–'.

Configuration	D0	D1	D2	D3	D4	D5
TopK SAE	1 ± 0	16 ± 3	811 ± 156	28 ± 41	–	–
EWG-SAE	1 ± 0	82 ± 2	1008 ± 71	6 ± 1	–	–
ArchMSAE	1 ± 0	41 ± 2	584 ± 45	14 ± 9	–	–
ActMSAE	1 ± 0	174 ± 17	2143 ± 109	539 ± 120	44 ± 12	4 ± 5
MPSAE	1 ± 0	1 ± 1	244 ± 215	–	–	–
H-SAE	1 ± 0	361 ± 0	5776 ± 0	–	–	–

this metric during graph creation; $\mathtt{s}_{ad}$ and $\mathtt{s}_{cc}$ are also weaker, showing that MCS-based graphs have generally worse activation behavior than CCA-based ones. Notably, MCS makes it possible to construct a graph for MPSAE (unlike CCA, whose low conditional activations prevent this). However, this MPSAE graph has near-random $\mathtt{s}_m$ and a near-zero $\mathtt{s}_{ha}$, indicating its hierarchy lacks semantic meaning, as children are, if anything, more abstract than their parents.

C.3 Validating Hierarchy Requirements with a Ground-Truth Hierarchy.

The requirements and metrics introduced in this work are intentionally designed to be evaluated unsupervised, without the need for detailed human hierarchy labels. While this makes them more applicable in many practical scenarios where ground-truth hierarchies are not known, it is still possible to validate the structure-level metrics against known ground-truth hierarchies. To do so, we use the well-known ImageNet-1k class hierarchy [14] and evaluate our metrics on images from ImageNet directly. We randomly sample up to 1000 images per class and use class membership as a binary neuron activation for the metric computations. For $\mathtt{s}_{ss}$, we use the average embedding of 20 randomly selected images per class as the "top-activating" images.

The results in Tab. 16 show a strong connection between the monosemanticity score and hierarchy level, indicated by an $\mathtt{s}_{ha}$ of nearly 1. This supports our claim that specificity (as measured by $\mathtt{s}_m$) increases through the hierarchy, and confirms that $\mathtt{s}_{ha}$ is a reasonable measure in this context. Further, the similarity between siblings in this hierarchy is captured by $\mathtt{s}_{ss}$.

C.4 Investigating the Connection between Sibling Similarity and Feature Redundancy

Sibling similarity itself does not check for near-duplicate features, since a feature is trivially most similar to itself. To further quantify how this metric relates to redundancy, we additionally report the Jaccard index [10] as a redundancy measure between siblings (cf. Tab. 17, left). This measures how often two siblings co-activate relative to how often at least one activates, and has also been used in the context of SAEs [38]. For around 1700 features of ActMSAE on CLIP and around 700 features on DINO, the Jaccard similarity to the closest sibling exceeds 0.5 and is potentially redundant. This occurs primarily at lower levels of the hierarchy, where increasingly similar features get grouped together (cf. Tab. 17, right).

Overall, while most SAE features remain quite distinct, features at lower depths become increasingly similar, making it harder to automatically distinguish highly similar from truly redundant features, which is an important general question for SAE feature spaces rather than a weakness specific to our approach.

Table 15: **Hierarchy metrics for graphs created with different similarity measures.** Conditional co-activations (CCA) and masked cosine similarity (MCS) on CLIP. Mean ± standard deviation over three seeds. No graph could be constructed for MPSAE with CCA because of its low conditional activations.

Condition	SAE	Nodes	$\mathtt{s}_m$	$\mathtt{s}_{ha}$	$\mathtt{s}_{ss}$	$\mathtt{s}_{rf}$	$\mathtt{s}_{ai}$	$\mathtt{s}_{ad}$	$\mathtt{s}_{cc}$
CCA	TopK SAE	2120 ± 42	0.616 ± 0.001	0.986 ± 0.003	0.750 ± 0.012	0.162 ± 0.008	0.914 ± 0.002	0.621 ± 0.023	0.257 ± 0.017
MCS	TopK SAE	1567 ± 93	0.564 ± 0.001	0.858 ± 0.024	0.766 ± 0.004	0.385 ± 0.011	0.587 ± 0.015	0.738 ± 0.004	0.301 ± 0.009
CCA	ArchMSAE	1012 ± 99	0.455 ± 0.000	0.332 ± 0.012	0.682 ± 0.012	0.215 ± 0.007	0.895 ± 0.002	0.907 ± 0.015	0.495 ± 0.013
MCS	ArchMSAE	2517 ± 15	0.441 ± 0.000	0.521 ± 0.017	0.700 ± 0.005	0.352 ± 0.008	0.324 ± 0.002	0.667 ± 0.003	0.369 ± 0.008
CCA	ActMSAE	3341 ± 19	0.598 ± 0.001	0.763 ± 0.011	0.800 ± 0.001	0.476 ± 0.007	0.940 ± 0.001	0.788 ± 0.009	0.487 ± 0.003
MCS	ActMSAE	4459 ± 19	0.567 ± 0.001	0.853 ± 0.001	0.722 ± 0.001	0.480 ± 0.004	0.154 ± 0.002	0.683 ± 0.008	0.368 ± 0.003
CCA	EWG-SAE	974 ± 57	0.530 ± 0.003	0.953 ± 0.002	0.624 ± 0.004	0.075 ± 0.003	0.918 ± 0.004	0.622 ± 0.009	0.202 ± 0.020
MCS	EWG-SAE	1368 ± 25	0.501 ± 0.003	0.681 ± 0.030	0.636 ± 0.008	0.258 ± 0.012	0.391 ± 0.012	0.531 ± 0.014	0.223 ± 0.008
CCA	MPSAE	—	—	—	—	—	—	—	—
MCS	MPSAE	504 ± 23	0.472 ± 0.003	0.030 ± 0.008	0.797 ± 0.013	0.398 ± 0.009	0.122 ± 0.003	0.942 ± 0.009	0.336 ± 0.008

Table 16: **Evaluating the ImageNet-1k hierarchy.** Applying our concept- and structure-level metrics to the ImageNet-1k hierarchy indicates a strong connection between the desired metrics and the ground-truth hierarchy structure.

Metric	$\mathtt{s}_m$	$\mathtt{s}_{ha}$	$\mathtt{s}_{ss}$
Mean	0.6855	0.9401	0.8639

C.5 Connection to Hierarchical Clustering Metrics

While metrics for hierarchical clustering are related to concept hierarchies, there are structural differences in the assumptions of the hierarchy elements. The basic elements of our feature hierarchy graph are features, compared to samples in a dendrogram. Intermediate nodes are also meaningful higher-level features, whereas dendrogram intermediate nodes are purely structural connection points. Further, our hierarchy graph is not a strict tree but a DAG with potentially disconnected components, where edges carry explicit semantic meaning rather than representing similarity-based merges.

These differences make a direct adaptation of hierarchical clustering metrics difficult and require a small adaptation of the cophenetic correlation to apply it to feature hierarchy graphs.

The original cophenetic correlation coefficient between a dissimilarity matrix x and the cophenetic distances t of a dendrogram is

$$c = \frac{\sum_{i<j}\big(x(i,j)-\bar{x}\big)\big(t(i,j)-\bar{t}\big)}{\sqrt{\sum_{i<j}\big(x(i,j)-\bar{x}\big)^2 \sum_{i<j}\big(t(i,j)-\bar{t}\big)^2}}, \tag{8}$$

where $x(i,j)$ is the original dissimilarity between leaves i and j, $t(i,j)$ is the cophenetic distance (the height of the dendrogram at which i and j are first merged), and $\bar{x}$, $\bar{t}$ are the means of $x(i,j)$ and $t(i,j)$ over all pairs $i<j$. In our adaptation, $t(i,j)$ is replaced by the depth of $\mathrm{LCA}(i,j)$ and $x(i,j)$ by the pairwise similarity like in $\mathtt{s}_{ss}$. Since all SAEs except ActMSAE have only two levels, this reduces to measuring whether more similar nodes share a parent or fall under different parents (where the LCA is the artificial root at depth 0), making it closely related to $\mathtt{s}_{\mathrm{ss}}$.

The results in Tab. 18 then also match our evaluation on CLIP: EWGSAE's and H-SAE's sibling similarities are close to random, while ActMSAE's is highest, and highlighting the connection between both metrics.

C.6 More Qualitative Illustrations

For the illustrative examples, we show a representative subset of child features and not necessarily all examples to their lowest depth to maintain readability.

Semantic failure modes of learned hierarchies. To complement the qualitative examples in the main paper (cf. Sec. 4), we provide additional examples here. While most ActMSAE hierarchy

Table 17: Sibling Jaccard similarity. **Left:** Average Jaccard similarity between siblings for each SAE type on CLIP ViT-L/14 and DINOv2-base. **Right:** Average Jaccard similarity to the closest sibling of ActMSAE features by hierarchy depth. (Values at depth ≥ 5 are unreliable due to low feature counts at that depth.)

SAE type	CLIP ViT-L/14	DINOv2-base
TopK SAE	0.024	0.095
EWG-SAE	0.008	0.012
ActMSAE	0.320	0.214
ArchMSAE	0.023	0.071
H-SAE	0.000	0.021

Depth	CLIP ViT-L/14	DINOv2-base
1	0.065	0.148
2	0.485	0.271
3	0.476	0.361
4	0.457	0.434
5	0.457	0.246
6	0.459	0.522

Table 18: Number of leaves and cophenetic correlation for each SAE type on CLIP.

	TopK SAE	EWG-SAE	ArchMSAE	ActMSAE	H-SAE
Leaves	1884	808	922	1971	5776
Cophenetic correlation	0.187	0.022	0.220	0.252	0.077

subgraphs are human-interpretable and semantically sensible, there are also cases where the learned hierarchies are harder to follow or lack clear meaning. Fig. 8 shows two such failure cases. In the left example, the automated naming strategy of Rao et al. [45] assigns the label *asheville* to a concept primarily centered around birds in flight against open sky. The children have only a loose semantic connection to the parent and low mutual similarity, covering concepts such as moon imagery, plants, and birds. The right example groups several concepts under *moroccan*, where most children are themselves noisy, consisting, for instance, of placeholder sprites. These two examples illustrate distinct failure modes: the first groups seemingly unrelated concepts based on spurious co-occurrences, while the second contains low-quality features that could have been filtered out by feature-interpretability evaluations prior to graph construction.

Illustration of H-SAE's Failure Mode. While H-SAE produces a fixed graph structure with consistent parent-child co-activations, our evaluation highlights a systematic lack of hierarchical semantics. Specifically, the hierarchical abstractness score $\mathtt{s}_{ha}$ reveals that parent features are typically more specific than their children — the opposite of what a generalization/specialization hierarchy requires. The underlying cause is that child features often fail to represent coherent semantic concepts, resulting in near-random $\mathtt{s}_m$ scores and consequently near-zero $\mathtt{s}_{ha}$. This is, therefore, less a failure of hierarchical structure per se than a general quality problem with the learned child features. Fig. 9 and Fig. 10 illustrate this: the parent concepts are somewhat coherent, environmental scenes and pollution in the first example, indoor kitchens and living rooms in the second, but the children bear no clear semantic connection to their parents. The top-activating images for the child features

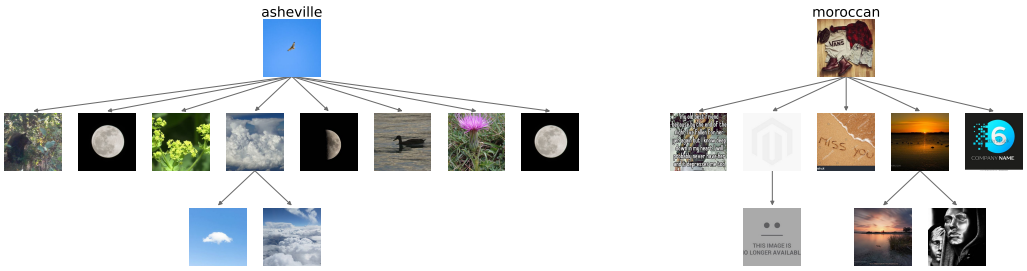
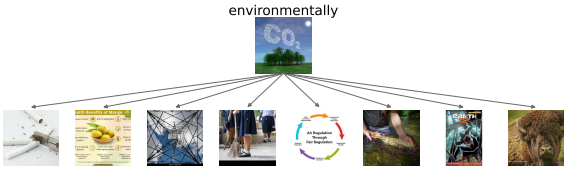


Figure 8: **Failure modes of learned hierarchies.** Left: children are grouped under a common parent based on spurious co-occurrences rather than clear semantic connections. Right: several child features are themselves noisy, for example, activating on placeholder sprites, which could be mitigated by feature-interpretability filtering prior to graph construction. All examples are taken from ActMSAE trained on CLIP.



(a) H-SAE Subgraph Example

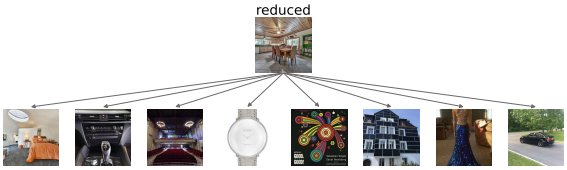


(b) Parent



(c) Leftmost child

Figure 9: **Qualitative example of H-SAE's failure mode.** The parent concept represents a coherent concept about the environment and its pollution, while the children do not have clear semantic connections to the parent. Even more, the top-activating images of the leftmost child themselves do not show any coherent concept.



(a) H-SAE Subgraph Example



(b) Parent



(c) Leftmost child

Figure 10: **Further qualitative example of H-SAE.** In this example, the parent concept represents a living room or a kitchen, while the children again do not have clear semantic connections to the parent. Similarly, the top-activating images of the leftmost child do not show a coherent concept.

themselves show no discernible underlying concept, underscoring that child-feature interpretability is the core limitation of this approach.

Analyzing the distributions of the results. In the main evaluation, hierarchy metrics are reported as means and standard deviations over a single graph. Here, we examine the distribution of individual scores within each graph in more detail. Fig. 11 and Fig. 12 show the results for CLIP and DINOv2 respectively. For most metrics and methods, the distributions are approximately unimodal and well-behaved. Two notable exceptions stand out, however. First, ActMSAE shows bimodal behavior across several metrics, likely reflecting the difference between hierarchy levels observed in Fig. 4: top-level graph elements tend to have lower $\mathtt{s}_m$ and lower $\mathtt{s}_{rf}$, pulling the distribution apart into two groups. Second, while the mean $\mathtt{s}_{cc}$ is above zero for all SAEs, each method has a substantial number of parent-child groups with negative conditional correlation, which means that parent activations decrease as child activations grow. This is precisely the soft absorption behavior our metrics are designed to capture, and its prevalence across methods underscores that it remains an unsolved challenge for current hierarchical SAE approaches.

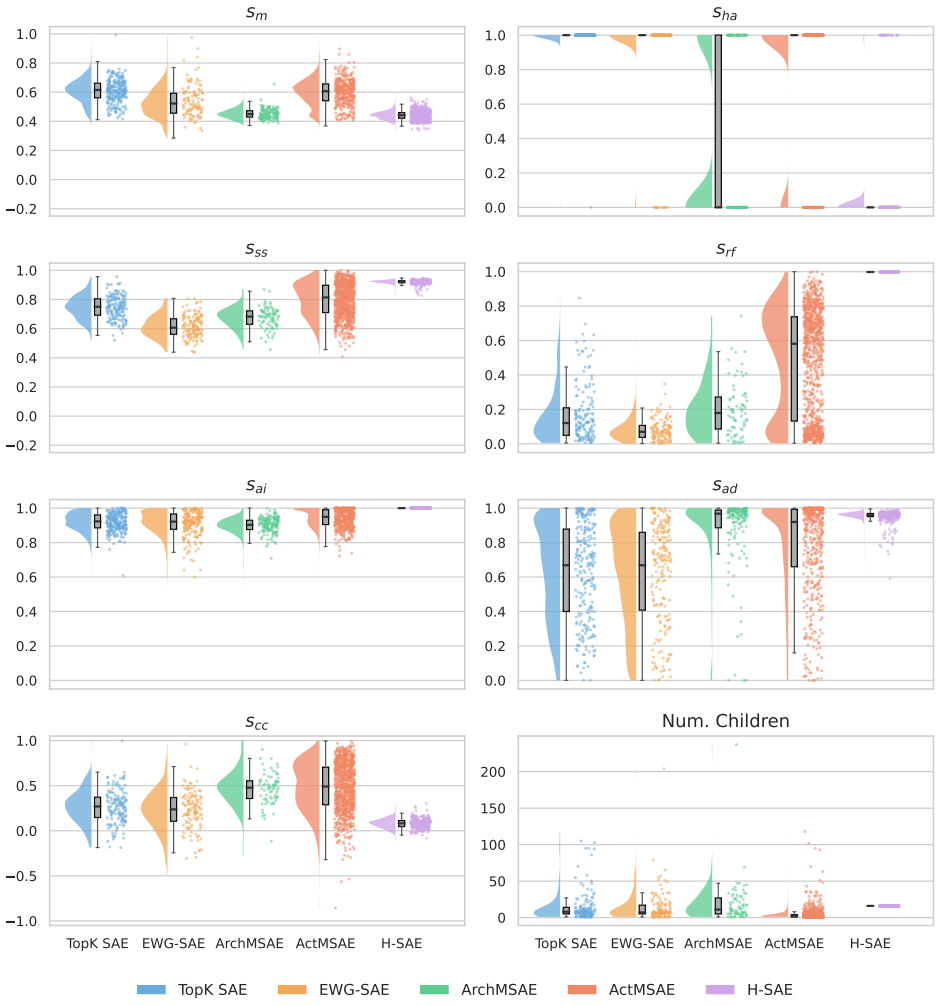


Figure 11: **Distribution of the metrics per graph element for SAEs trained on CLIP.** While the distributions of nodes, edges, or parent-child groups are mostly well-behaved, there is a clear two-modal distribution for ActMSAE visible, most likely due to the big differences between hierarchy depths. ($\mathtt{s}_{ha}$ takes only discrete values, meaning that the raincloud plots for this metric do not provide more information than the mean).

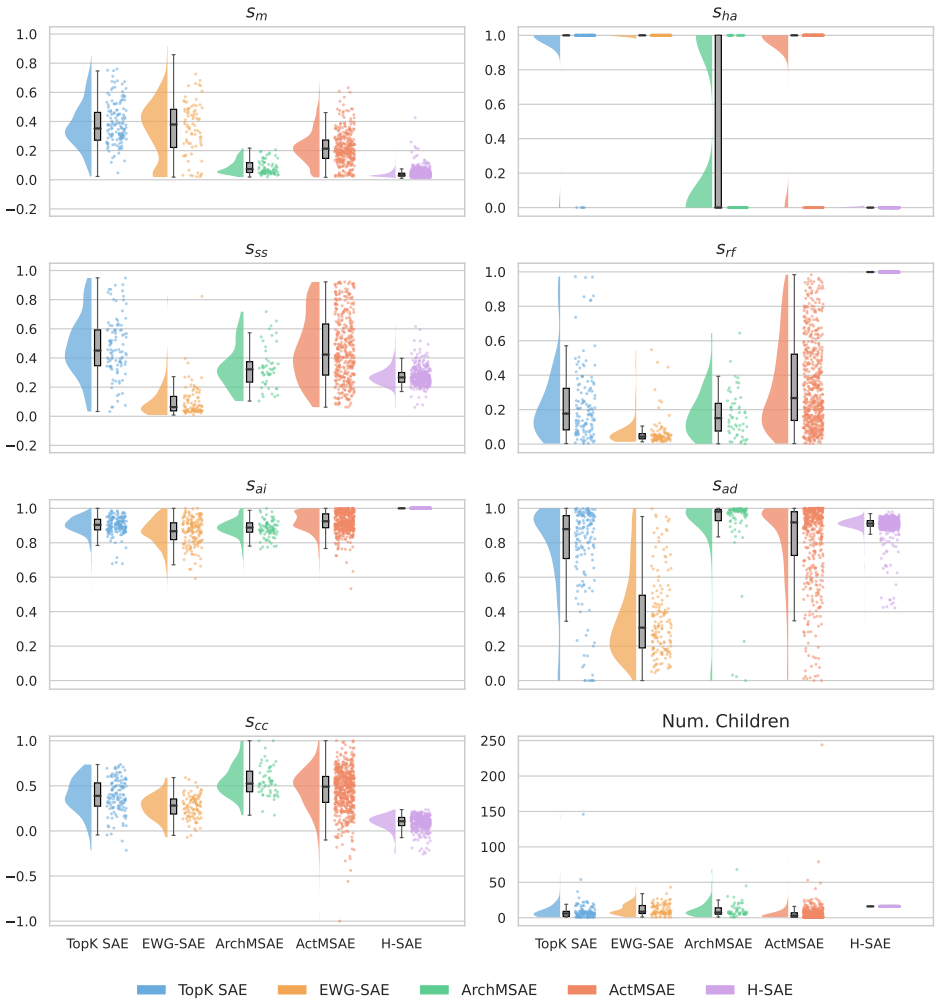


Figure 12: **Distribution of the metrics per graph element for SAEs trained on DINOv2.**

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: The abstract states that the in this work, requirements for concept hierarchies and based on these concrete metrics for SAEs are developed (these are provided in Sec. 3). Then the abstract states that current hierarchical SAE approaches are evaluated on visual data based on their hierarchies, which is done in Sec. 4 and the main findings (e.g., hard and soft absorption) are discussed in Sec. 5.

   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: We discuss central limitations in a separate part of the discussion section Sec. 5.

   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: [N/A]

Justification: The paper does not include theoretical results.

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: The hyperparameters and training details for all SAE methods are provided in the appendix Sec. B.3. The details for the graph creation method are provided in the appendix Sec. B.4. Additionally, there the code is made available.

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: The used dataset (CC3M) is openly available. The base encoders (CLIP and DINOv2) are also openly available and the specific versions used are specified. The code for the evaluation and training of the hierarchical SAEs is provided with the paper.

   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: The details for SAE training and graph creation are provided in the appendix (Sec. B.3 and Sec. B.4).

   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: All experiments include standard deviations.

   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: The compute ressoures required to run the experiments are stated in Sec. A.2.

   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: The paper does not include any experiments involving human subjects nor any new datasets. The impact and implictation of the work are dicussed in the remaining checklist and the broader impact section.

   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: [Yes]

    Justification: The broader impact of this work is discussed in Sec. A.1.

    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: The paper does not include released models or datasets

    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: All used assets are cited (dataset, base vision encoders and SAE approaches). The CC3M dataset is freely usable.

    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: [N/A]

    Justification: The paper does not include new assets.

    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: The paper does not include any crowdsourcing/research with 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 paper does not include any crowdsourcing/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: [N/A]

    Justification: The main methodological contribution (requirements and metrics for concept hierarchies) do not involve LLMs.

    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.